DECISION-THEORETIC PLANNING UNDER RISK-SENSITIVE PLANNING OBJECTIVES

A Dissertation
Presented to
The Academic Faculty

by

Yaxin Liu
刘亚新

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

College of Computing
Georgia Institute of Technology
April 2005

DECISION-THEORETIC PLANNING UNDER RISK-SENSITIVE PLANNING OBJECTIVES

Approved by:

Dr. Sven Koenig, Co-advisor
College of Computing
Georgia Institute of Technology

Dr. Craig Tovey, Co-advisor
College of Computing
Georgia Institute of Technology

Dr. Anton Kleywegt
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Frank Dellaert
College of Computing
Georgia Institute of Technology

Dr. Ashok Goel
College of Computing
Georgia Institute of Technology

Dr. Richard Goodwin
IBM T.J. Watson Research Center

Date Approved: April 11, 2005

To Zhijie and Emily.

Hofstadter's Law: It always takes longer than you think, even when you take into account Hofstadter's Law.

— *Douglas R. Hofstadter*

ACKNOWLEDGEMENTS

Many thanks to my advisor, Sven Koenig, for helping me through the process. Thanks to my co-advisor, Craig Tovey, for many encouraging conversations. Thanks to Richard Goodwin, for his guidance of my summer internships at IBM and for shaping the direction of this work. Thanks to Anton Kleywegt, for his encouragement and support from early on. Thanks to Frank Dellaert, for his advice on writing. And thanks to Ashok Goel, for helping in various ways.

Thanks to my fellow students, David Furcy, for long-term encouragement and collaboration; to Jimmy Davies, for moral support; to Chip Mappus, for much help with English; and to Patrick Yaner, for his bookshelf.

Thanks to Peter Stone, for allowing me to finish the thesis at UT Austin.

Special thanks to my friends Yuan Chen and Xiaoling Cao, for helping taking care of Emily.

I am forever in debt to Zhijie for the many years of enduring, and to Emily for not-so-many years. Lastly and not leastly, I appreciate the help and support from my parents and parents-in-law.

This work is partially supported by an IBM Fellowship.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

SUMMARY

Risk attitudes crucially affect human decision making preferences, especially in scenarios where huge wins or losses are possible, as exemplified by planetary rover navigation, oilspill response, and business applications. Decision-theoretic planners therefore need to take risk aspects into account to serve their users better. However, most existing decision-theoretic planners use simplistic planning objectives that are risk-neutral. The thesis research is the first comprehensive study of how to incorporate risk attitudes into decision-theoretic planners and solve large-scale planning problems represented as Markov decision process models. The thesis consists of three parts.

The first part of the thesis studies risk-sensitive planning in the case where exponential utility functions are used to model risk attitudes. In this case, there exists an optimal plan that maps states to actions. I show that existing decision-theoretic planners can be transformed to take risk attitudes into account. My approach is more general than previous approaches, which transform the planning tasks rather than the planning algorithms. The transformed algorithms bear visual resemblance to the original algorithms but special treatment may be needed to ensure their validity. Moreover, different transformations are needed if the transition probabilities are implicitly given, namely, temporally extended probabilities and probabilities given in a factored form. I show how the transformations and their variants can be applied to various decision-theoretic planners.

The second part of the thesis studies risk-sensitive planning in the case where general nonlinear utility functions are used to model risk attitudes. In this case, there does not in general exist an optimal plan that maps states to actions, and an optimal plan must take into consideration the accumulated rewards starting from the initial state. I show that a state-augmentation approach can be used to reduce a risk-sensitive planning problem to a risk-neutral planning problem with an augmented state space. I further use a functional interpretation of value functions and approximation methods to solve the planning problems efficiently with value iteration. I also develop an exact method for solving risk-sensitive planning problems where one-switch utility functions are used to model risk attitudes.

The third part of the thesis studies risk sensitive planning in case where arbitrary rewards are used. In this case, many of the basic properties are unknown, including the existence and finiteness of optimal expected utilities. I propose a spectrum of conditions that can be used to constrain the utility function and the planning problem so that the optimal expected utilities exist and are finite. These conditions are the basis for the further development of computational procedures. I prove that the existence and finiteness properties hold for stationary plans, where the action to perform in each state does not change over time, under different sets of conditions.

I use two running examples to demonstrate that risk-sensitive planners can be easily created from their risk-neutral counterparts, and the resulting optimal plans are qualitatively different from optimal plans under a risk-neutral planning objective.

CHAPTER I

INTRODUCTION

In the not-very-distant future, software agents will be able to provide decision support for human decision makers, or even behave autonomously in the environment on behalf of human decision makers. These software agents need built-in planning capabilities to cope with the complexity of real-world applications, because decisions are not made in isolation and earlier decisions will affect how later decisions can be made. Artificial intelligence (AI) planning deals with the problem how such decisions can be made. It is important that planners of such agents have the same planning objectives as their human users; otherwise the plans generated by them are not what people really want, and thus of little use to their human users. Planners of such agents also need to deal with uncertainty, which, for example, can result from sensor or actuator errors, as well as incomplete information or incomplete modeling of the environment. However, current decision-theoretic (DT) planners, although a promising approach to planning under uncertainty, are insufficient for such real-world applications since they use simple planning objectives that often do not meet people's needs.

A human decision maker's preference structure indicates how he or she compares different outcomes of his or her decisions. The preference structure determines the planning objective. Decision theory has studied empirical and normative preference structures of human decision makers. However, it only specifies what optimal plans should be, but not how to obtain them efficiently. On the other hand, decision-theoretic planning has developed planners that solve artificial intelligence planning problems efficiently by exploiting structures of these planning problems. But it only uses simple planning objectives such as maximizing the expected plan-execution reward or maximizing the probability of achieving a goal state. Therefore, the current "decision-theoretic" planning research is not complete,

Table 1.1: An illustrative example for risk attitudes

	Probability	Reward	Expected Reward	Utility	Expected Utility
Alternative 1	50%	$10,000,000	$5,000,000	0.95	0.475
	50%	$ 0		0.00	
Alternative 2	100%	$ 4,500,000	$4,500,000	0.74	0.740

without investigating how to plan efficiently with more realistic preference structures from decision theory.

This thesis studies risk-sensitive planning objectives, an important type of realistic planning objectives, in the decision-theoretic planning framework, by combining principles from decision theory, theoretical foundations from operations research (OR), and constructive methods from artificial intelligence planning, thus combining the strengths of these decision-making disciplines and extending the applicability of AI planners.

1.1 Risk Attitudes

Risk-sensitive planning objectives are planning objectives that take into account human users' risk attitudes. Risk attitudes are an important type of preference structure that influences how people make decisions in domains where huge wins or losses are possible. Many domains in AI planning under uncertainty are such high-stake planning domains since huge losses of money, equipment, or even human life are possible in these domains. Examples include space applications, such as autonomous spacecraft control (Pell *et al.*, 1998) and rover navigation (Simmons *et al.*, 1995; Zilberstein *et al.*, 2002); environmental applications, such as fighting forest fires (Cohn *et al.*, 1989) and containing marine oil spills (Blythe, 1997); and business applications, such as production planning (Murthy *et al.*, 1999) and autonomous trading agents (Goodwin *et al.*, 2002).

In high-stake decision scenarios, human decision makers usually do not maximize the expected reward because they take their risk attitudes into account. Risk attitudes explain why human decision makers buy insurance even though the insurance premium is usually much larger than the expected loss from the insurance cause, and also why human decision makers buy lottery tickets even though the ticket price is usually much larger than the expected lottery prize. Before considering how risk attitudes affect planning, we first illustrate the concept of risk attitudes more specifically using a simple lottery example shown in Table 1.1. The decision problem involves a choice between two alternatives, and one can choose to participate in one and only one of the two lotteries at no charge. When human decision makers have to decide whether they would like to choose Alternative 1: \$10,000,000 with 50% probability (and nothing otherwise), or Alternative 2: \$4,500,000 for sure, many prefer Alternative 2 although its expected reward is clearly lower — they are risk-averse (basically meaning that they tolerate a smaller expected reward for a reduced variance, and thus put more weights on the worst case of all possible outcomes, although this explanation is a bit simplified). If a planner chooses Alternative 1, then many human decision makers will be extremely unhappy half of the time. But this is exactly what a planner will do under the currently popular planning objective in decision-theoretic planning research, which maximizes the expected reward.

Decision theory (French, 1986) has been used as a normative framework for making rational decisions under uncertainty, and can explain why the behavior of choosing Alternative 2 is perfectly rational. The two components of decision theory are probability theory and utility theory, where probability theory deals with uncertainty and utility theory deals with preferences. Utility theory (von Neumann and Morgenstern, 1944; Fishburn, 1970; Barberá *et al.*, 1998) suggests that people prefer an alternative with the maximal expected utility, where the utility, in this context, is the value of a monotonically increasing function of the wealth level (how much money one has), and the function is risk-sensitive if it is nonlinear. Intuitively, the utility reflects how happy the decision maker is with its current wealth level. Maximizing the expected utility and maximizing the expected reward result in the same decisions if either the domain is deterministic or the utility function is linear, in which case the utility function and the agent are referred to as risk-neutral. These assumptions, however, are often not satisfied. For example, nonlinear utility functions are necessary to account for the risk-averse attitudes of many human decision makers in the lottery example above.

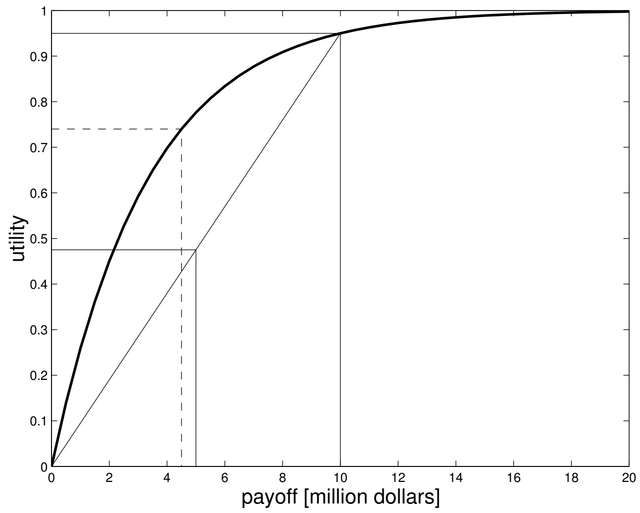
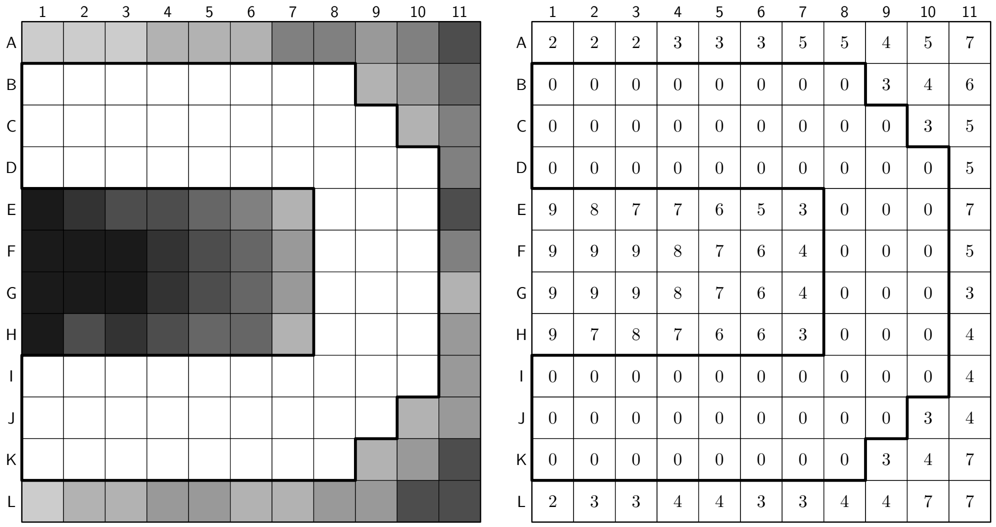


Figure 1.1: Interpretation of the lottery example using a concave exponential utility function

Utility theory explains the behavior of taking Alternative 2 as follows. Exponential utility functions are a popular type of risk-sensitive utility function (Corner and Corner, 1995). With a concave exponential utility function $1 - 0.9999997^w$ when being offered the lotteries, the human decision maker associates utility 0.00 with a wealth level of \$0, utility 0.74 with a wealth level of \$4,500,000, and utility 0.95 with a wealth level of \$10,000,000. Then, the (expected) utility of getting \$4,500,000 for sure is 0.74, whereas the expected utility of getting \$10,000,000 with 50% probability is only 0.475. These calculations are also depicted in Figure 1.1, where the solid lines correspond to Alternative 1 and the dashed lines correspond to Alternative 2. In this case, Alternative 2 maximizes the expected utility for this person, which captures why this person prefers the safe alternative (Alternative 2) over the one with the larger expected reward (Alternative 1). Other human decision makers can have other utility functions and thus may arrive at different decisions. For example, some human decision makers, such as aggressive gamblers, can be risk-seeking, who would welcome a larger variance for a reduced expected reward, and thus put more weight on the best case of all possible outcomes. For example, a risk-seeking person may still take Alternative 1 in the lottery example for the chance of winning big, even if the sure reward



(a) The Map

	1	2	3	4	5	6	7	8	9	10	11
A	2	2	2	3	3	3	5	5	4	5	7
B	0	0	0	0	0	0	0	0	3	4	6
C	0	0	0	0	0	0	0	0	0	3	5
D	0	0	0	0	0	0	0	0	0	0	5
E	9	8	7	7	6	5	3	0	0	0	7
F	9	9	9	8	7	6	4	0	0	0	5
G	9	9	9	8	7	6	4	0	0	0	3
H	9	7	8	7	6	6	3	0	0	0	4
I	0	0	0	0	0	0	0	0	0	0	4
J	0	0	0	0	0	0	0	0	0	3	4
K	0	0	0	0	0	0	0	0	3	4	7
L	2	3	3	4	4	3	3	4	4	7	7

(b) Costs of States

Figure 1.2: A grid-world robot navigation example

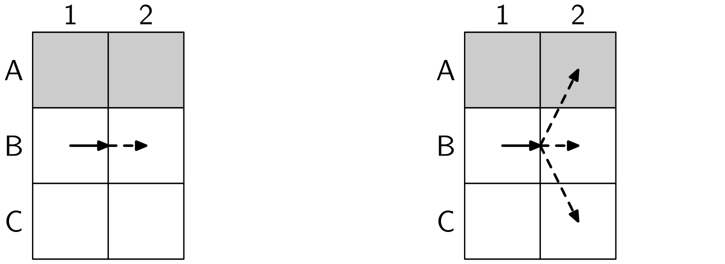
in Alternative 2 were increased to \$5,500,000, which would be higher than the expected reward from Alternative 1.

Intuitively, risk attitudes will also affect which plan to choose in high-stake planning domains, where the multiple decisions are made sequentially. In this thesis, I study how to incorporate risk attitudes into decision-theoretic planning to make it more suitable for building software agents in real-world applications. Before we discuss risk-sensitive planning in more detail, we first briefly review some concepts from AI planning and DT planning, and introduce two running examples.

1.2 AI Planning and Two Examples

In this section, we briefly review some concepts of AI planning under uncertainty. This review is necessary since we need to re-examine the concepts from current decision-theoretic planning under risk-sensitive planning objectives, in order to understand their similarities and differences.

We also introduce two running examples, a robot navigation problem and a painted blocks problem, which will be used throughout this thesis. The robot navigation example is also used in this section to illustrate concepts in AI planning under uncertainty. The painted blocks example will be introduced toward the end of this section.

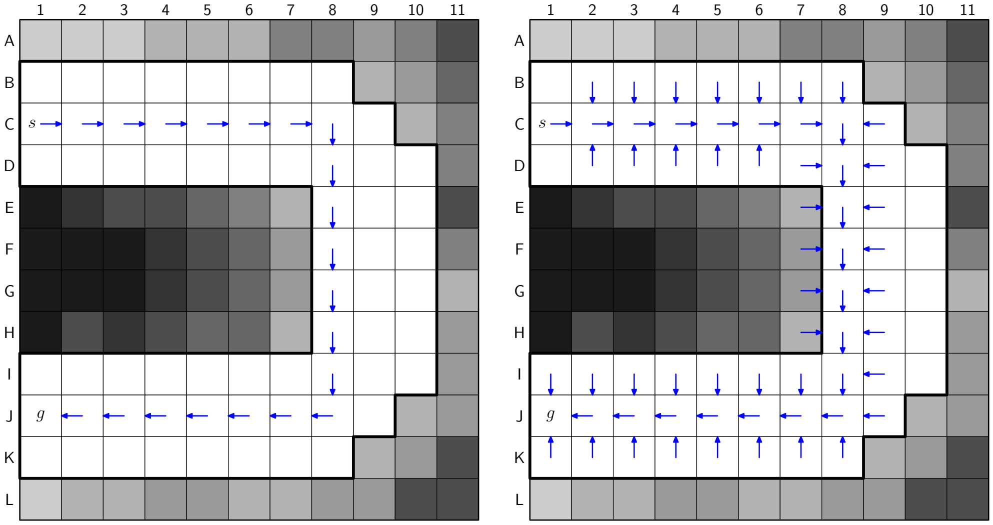


(a) A Deterministic Action (b) A Nondeterministic Action

Figure 1.3: Action models

Planning in artificial intelligence concerns determining how intelligent agents should act in the environment. The agent can perform actions that change its states in the environment. The first example is a grid-world robot navigation problem. Figure 1.2(a) shows the map for this problem, where the states are cells, and actions are movements in the four main compass directions, north (`N`), east (`E`), south (`S`), or west (`W`). The robot is able to enter any cell in the map. If there is no uncertainty, the robot will move deterministically into the cell next to its current cell in the direction corresponding to the action it chose, as shown in Figure 1.3(a), where the solid arrow indicates the action and the dashed arrow indicates the resulting state. But if uncertainty is present, a movement action can have multiple distinct outcomes. Figure 1.3(b) illustrates such a case. A moving east action (`E`) in state `B1` can result in one of the states `A2`, `B2`, or `C2`. Usually actions are associated with costs or rewards. In the example, performing an action incurs a unit cost, and additional costs are possible if the robot enters muddy terrain (darker colors) but there are no additional costs if it remains on the road (white color). The additional costs of states are shown in Figure 1.2(b). The robot does not move when it bumps into the border of the grid world.

The agent usually needs to determine a series of actions that achieve a certain goal, and this process is called planning. A plan is a collection of actions that can be executed by the agent. If deterministic actions are sufficient to model the problem, the robot will be able to reach the goal state `J1` starting from the initial state `C1`, following a deterministic plan as shown in Figure 1.4(a). But if uncertainty is unavoidable, as in many real-world applications, the agent needs plans that can handle all possible contingencies. Such a plan is called a contingent plan. A contingent plan for the robot navigation example is



(a) A Deterministic Plan (b) A Contingent Plan

Figure 1.4: Deterministic and contingent plans

shown in Figure 1.4(b). This contingent plan tries to recover from unintended outcomes of actions by always going back to the nominal path suggested by the deterministic plan from Figure 1.4(a).

A trajectory is the course of actions and states resulting from executing a plan. Since we consider planning problems involving uncertainty, where there can be multiple distinct outcomes when performing the same action in the same state but at different times, executing a plan several times can result in multiple distinct trajectories. For example, Figure 1.5 shows two possible trajectories for executing the contingent plan from Figure 1.4(b).

Planning objectives concern how the agent chooses among plans based on the agent's preference structure. For example, the agent may prefer a plan achieving a goal state over one not achieving a goal state, or the agent may prefer a plan with a lower cost when achieving a goal state. For an autonomous agent acting on behalf of a human user, or a decision support agent for a human user, it is essential that the software agent uses the same preference structure as its human user. Otherwise the resulting plans may not be very useful for the user. When uncertainties are involved, the planning objectives should also take into account all contingencies, rather than just comparing plans based on one sample trajectory resulting from each plan. For example, in the robot navigation problem, the contingent plan shown in Figure 1.4(b) is often preferred over the deterministic plan shown

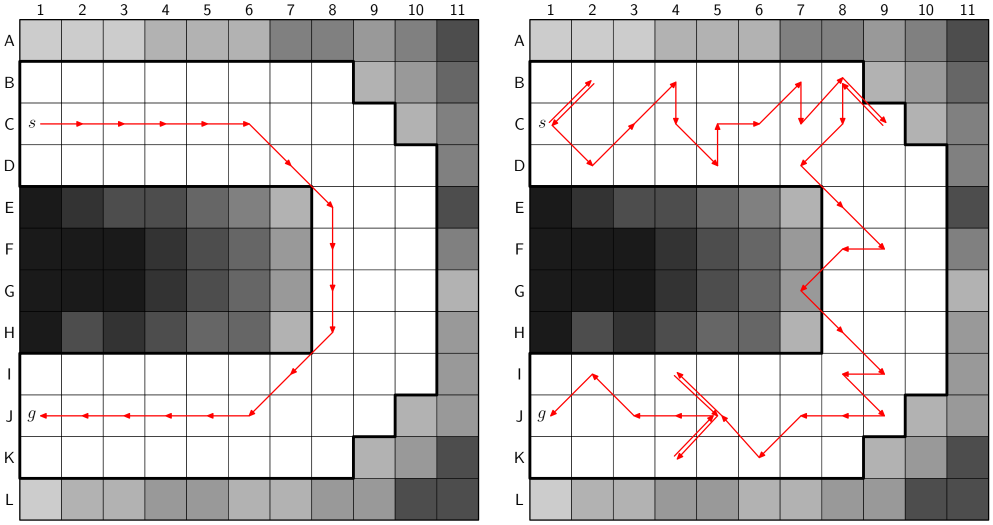
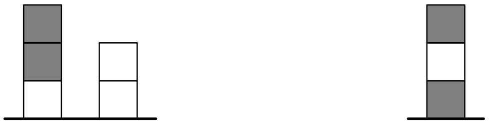


Figure 1.5: Two possible trajectories under the contingent plan

in Figure 1.4(a), since the former can deal with all possible trajectories, but the latter only takes into account one possible trajectory.

The second example is the painted blocks domain (Koenig and Simmons, 1994b). An agent tries to build towers of blocks. The agent can only move one top block at a time, and it can move the block onto the table or onto the top of another tower. We assume that the moving actions have uncertain outcomes. When the agent tries to move a top block onto the top of a tower, this block may drop onto the table, and thus the action fails. The action of moving a top block onto the table, however, always succeeds. Moreover, the agent can perform deterministic painting actions that paint a block black or white. We assume that it takes one minute to perform the moving action, and three minutes to perform the painting action. A planning problem in the painted-blocks domain is to convert a given initial configuration to a given goal configuration, which may only be partially specified. An example of a painted-blocks problem is shown in Figure 1.6. If we do not distinguish the individual blocks, the initial configuration Figure 1.6(a) is also the initial state, but the goal configuration Figure 1.6(b) corresponds to seven goal states since the remaining two blocks are not specified in the goal configuration.



(a) Initial Configuration (b) Goal Configuration

Figure 1.6: A painted blocks problem

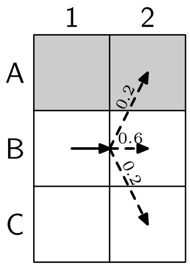


Figure 1.7: A probabilistic action model

1.3 Decision-Theoretical Planning

Decision-theoretic planning (Blythe, 1999; Boutilier *et al.*, 1999) deals with planning problems under uncertainty in a quantitative fashion. It uses probabilities to describe the likelihoods of uncertain outcomes of actions. For the robot navigation problem, we may assign probabilities as shown in Figure 1.7. For the painted blocks problem, we assume that the action of moving a block to the top of a tower succeeds with probability 0.5, which is different from (Koenig and Simmons, 1994b). Therefore, following a certain plan does not produce a single trajectory, but a probability distribution over possible trajectories. Since some trajectories are more preferable than others, the agent needs to trade off between the probabilities of realizing a trajectory and the preferability of doing so. For the robot navigation example, we can use simulation to illustrate the distribution of trajectories under the contingent plan from Figure 1.4(b) with the probabilistic action model from Figure 1.7. Since it is difficult to show the distribution of trajectories directly, we show the frequencies of state visits (Figure 1.8, where a darker cell indicates a higher probability) and total rewards (Figure 1.9) as indications of the distribution of trajectories. These results are obtained by simulating complete runs from the initial state to the goal state 200,000 times. For the contingent plan, the expected total reward is −30.74 and the variance of the total reward is 25.75.

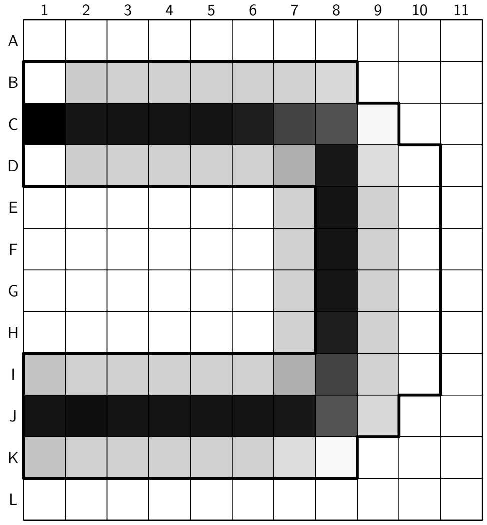


Figure 1.8: Simulated frequencies of state visits for the contingent plan

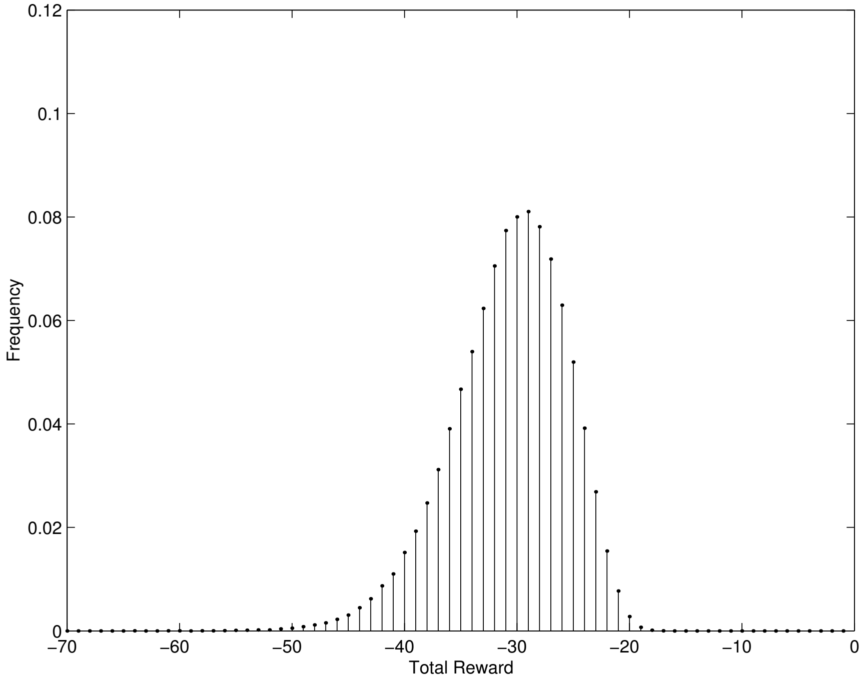


Figure 1.9: Simulated frequencies of total rewards for the contingent plan

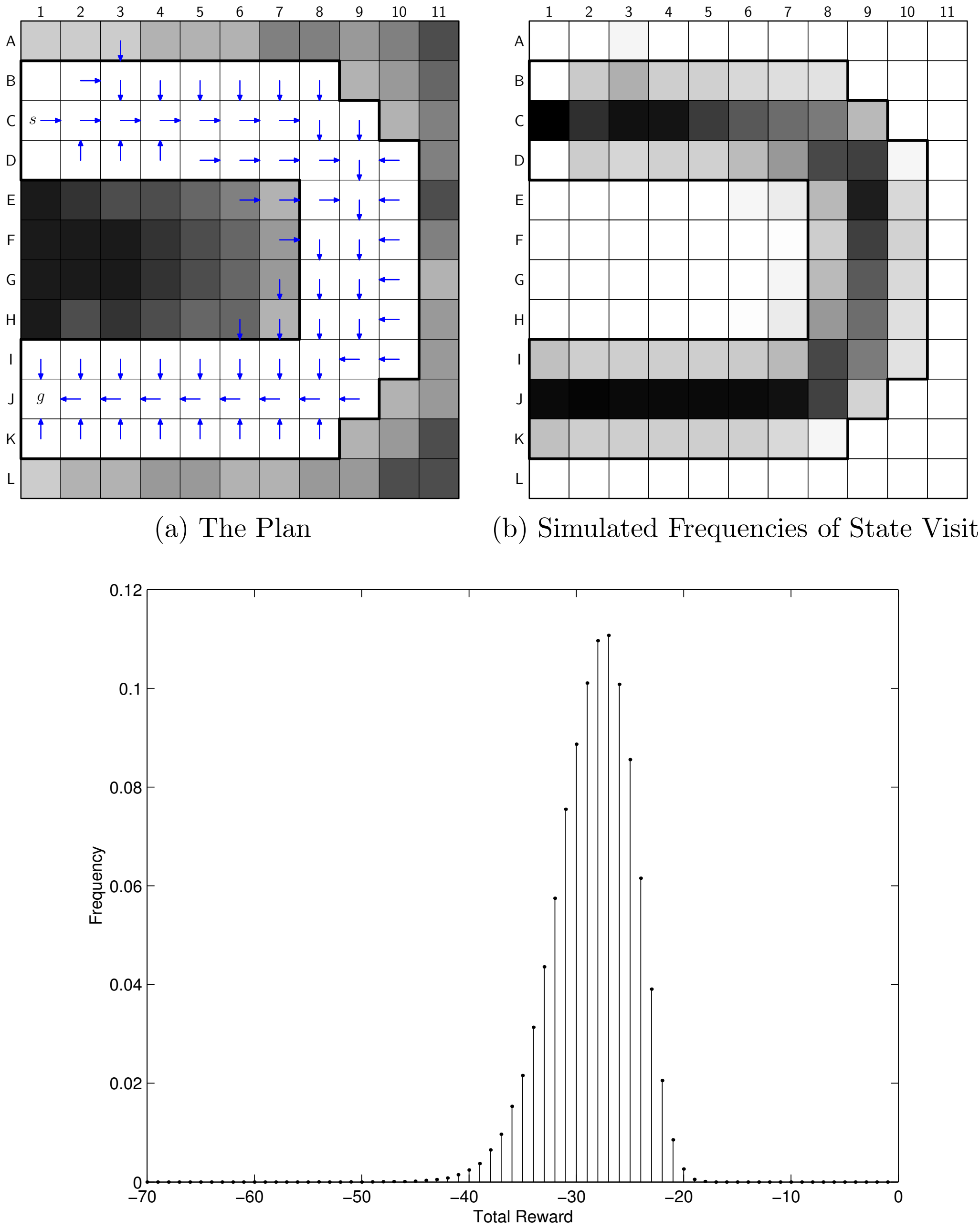
I use decision theory, especially utility theory, as a normative framework for making this tradeoff. In the context of planning, utility theory claims that under a set of reasonable assumptions, we can define a utility function that maps trajectories to real-valued utilities. The utility function is based on the agent's preference structure so that a trajectory with a higher utility is preferred over a trajectory with a lower utility. Moreover, the utility function assigns values such that when facing uncertainty, a rational agent's planning objective is to choose the plan that maximizes the expected utility (MEU) over all possible trajectories. Human decision makers sometimes deviate from utility theory because utility theory does not model human inadequacies in decision making and thus is not able to explain all empirical findings about human decision making (Kahneman *et al.*, 1982; Bell *et al.*, 1988). This is not a problem for planning, since planners are supposed to follow a theory of normative rather than empirical decision making in order to eliminate inconsistencies. Besides, recent empirical studies show that if human decision makers are given opportunities to learn their inconsistencies, their behaviors will converge to what utility theory suggests (van de Kuilen and Wakker, 2004). In this work, we assume that the agent is rational and follows the implications of utility theory, that is, it prefers the plan that maximizes the expected utility, where the expectation is taken over the probability space of the trajectories. When the maximizing plan is hard to find, the planning objective is relaxed to finding a plan that approximately maximizes the expected utility, that is, a "good" plan, which often can be obtained much faster.

However, utility theory by itself only provides methods for choosing actions in simple problems where all possible alternatives can be enumerated. For AI planning problems, there are too many possible plans to enumerate (in fact there can be an infinite number of plans). To alleviate this problem, many decision-theoretic planners adopt Markov decision process (MDP) models from operations research as the formal representation of planning problems under uncertainty (Koenig, 1992; Boutilier *et al.*, 1999). The MDP formulation can model various kinds of uncertainties such as uncertain outcomes of actions, exogenous events, and has been extended to partially observable MDPs (POMDPs) to model sensor uncertainty (Sondik, 1971). With the MDP formulation, planning problems involving uncertainty are now expressed in a language well-known to the operation research community and with a rich literature (Puterman, 1994; Feinberg and Shwartz, 2002), from which AI planning can use a lot of results and methods.

A popular planning objective for MDPs is to maximize the expected total reward (MER). This planning objective is based on a simple utility function that is defined as the total reward along the trajectory, by summing up the rewards obtained from each action. As suggested by utility theory, the agent will choose a plan that maximizes the expected total reward, which is a risk-neutral planning objective since the utility is identical to the total reward and the identity utility function is risk-neutral. For the robot navigation problem, an optimal plan under the MER objective is shown in Figure 1.10(a). The expected total reward is -28.33 and the variance of the total reward is 13.68. To compare it with the plan from Figure 1.4(b), we show the simulated frequencies of state visits in Figure 1.10(b) and total rewards in Figure 1.10(c).

Risk-neutral planning objectives[1] are used by most decision-theoretic planners. Under a risk-neutral objective, we can construct efficient AI planning algorithms based on a synthesis of results from operation research, probabilistic reasoning, and classical AI planning. For example, decision-theoretic planners from (Barto *et al.*, 1995; Hansen and Zilberstein, 2001) use heuristic search methods, which are popular in artificial intelligence (Hart *et al.*, 1968; Korf, 1990; Nilsson, 1980) and in AI planners for deterministic planning problems (Korf, 1987; Bonet and Geffner, 2001; Hoffman and Nebel, 2001), to focus computational effort on parts of the state space that are relevant for solving the planning problem; decision-theoretic planners from (Hauskrecht *et al.*, 1998; Sutton *et al.*, 1999b; Dietterich, 2000; Parr, 1998) use domain knowledge to hierarchically decompose planning problems, analogous to the hierarchical planning ideas in classical AI planners such as (Tate, 1977; Nau *et al.*, 1999);

[1] There are variants of the MER objective that we just described, for example, the discounted version MER_β. We refer to the MER objective and its variants collectively as risk-neutral planning objectives. They are described in more detail in Section 2.3.



(a) The Plan

(b) Simulated Frequencies of State Visits

(c) Simulated Frequencies of Total Rewards

Figure 1.10: An optimal plan under the MER objective

and decision-theoretical planners from (Boutilier *et al.*, 2000; Hoey *et al.*, 1999) use feature-based representations of the state space, similar to the representations used in classical AI planning research (Fikes and Nilsson, 1971; Pednault, 1989).

Nevertheless, risk-neutral objectives also have their negative side. They emphasize a particular utility function, and thus a particular preference structure. This is overly simplistic in some applications such as high-stake decision scenarios. To account for more realistic preference structures of human decision makers, we need to use the more general MEU objectives with more general utility functions. Since it is important to take human decision makers' risk attitudes into account when solving planning problems involving uncertainty, this thesis discuss general approaches and specific planners that plan under risk-sensitive planning objectives that reflect different risk attitudes.

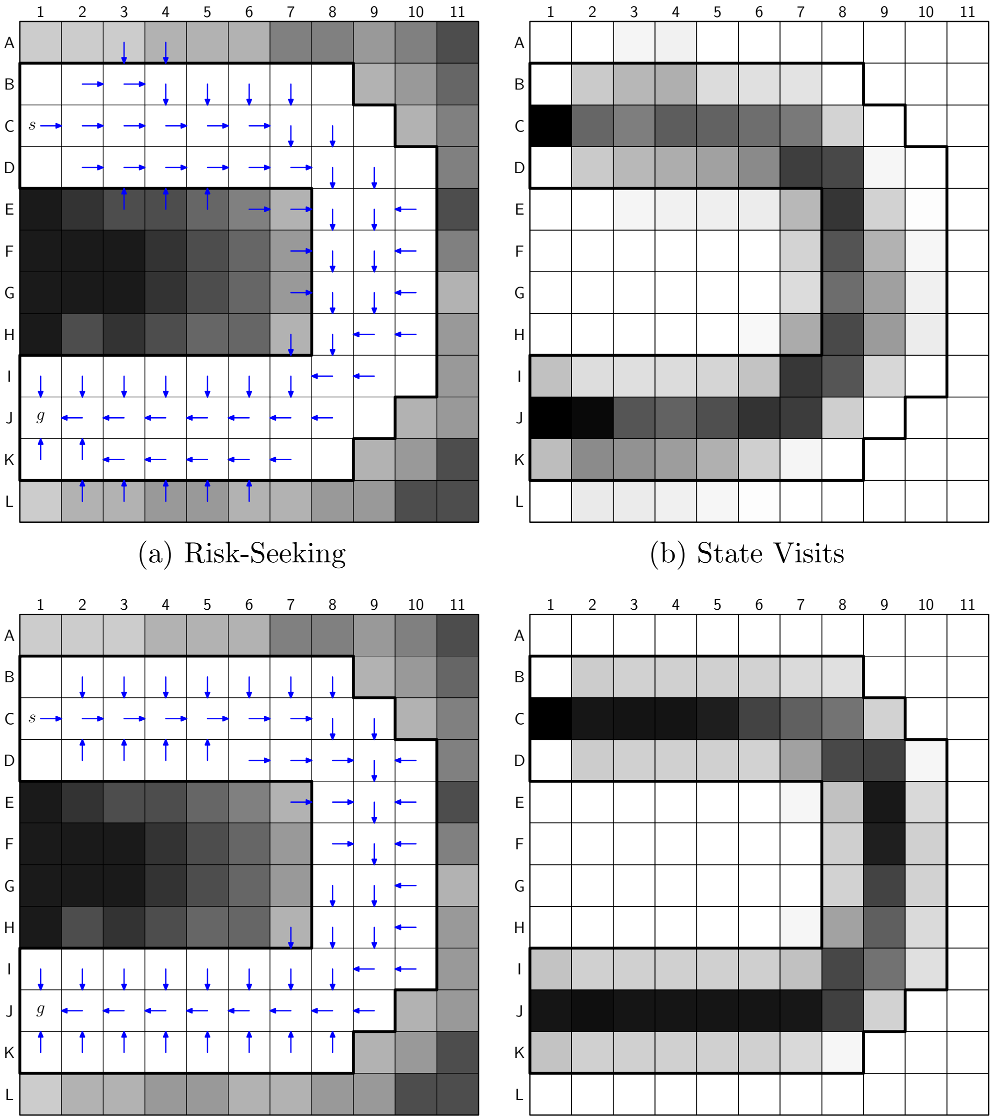
1.4 Risk-Sensitive Planning

As illustrated by the lottery example, the MER objective is often too simplistic to model the preference structures of human decision makers adequately. In particular, it cannot model the risk attitudes when facing high-stake decision scenarios.

How to plan under risk-sensitive planning objectives that are useful in high-stake scenarios, however, is a topic that has been neglected in the literature of AI planning, with very few exceptions (Koenig and Simmons, 1994a,b; Koenig, 1997; Koenig and Liu, 1999). It is nevertheless an important topic because the recommendations of planners should reflect the opinions of their users correctly, and risk attitudes are a common type of preference structure that cannot be captured by the overly simplistic risk-neutral objectives. Therefore, I believe that risk-sensitive planning has its place in AI planning research and will have its appropriate applications.

The results from risk-sensitive planners are qualitatively different from those from risk-neutral planners. This phenomenon reflects the effects of the risk aspects of human preference structures. We can demonstrate such effects with the robot navigation example. Suppose exponential utility functions are used. An optimal plan for a risk-seeking agent[2] is

[2] $\gamma = 1.4$ is a risk parameter. See Chapter 3 for the meaning of γ.



(a) Risk-Seeking

(b) State Visits

(c) Risk-Averse

(d) State Visits

Figure 1.11: Optimal plans under the MEU objective with exponential utility functions

shown in Figure 1.11(a), and the simulated frequencies resulting from this plan is shown in Figure 1.11(b). An optimal plan for a risk-averse agent[3] is shown in Figure 1.11(c), and the simulated frequencies resulting from this plan is shown in Figure 1.11(d). Comparing these two plans to the optimal plan under the MER objective (-28.33 and 13.68 respectively), we can identify qualitative differences between a risk-sensitive plan and a risk-neutral plan.

We can also interpret risk-sensitive planning objectives as a way of making tradeoffs between the mean and the variance of the total plan-execution reward. A risk-averse agent will tolerate a decrease in the mean in exchange for a smaller variance, while a risk-seeking agent will tolerate a decrease in the mean in exchange for a larger variance. For the robot navigation example, the risk-seeking optimal plan has an expected total reward of -29.89 and a variance of 38.16, and the risk-averse optimal plan has an expected reward of -28.66 and a variance of 12.34. Comparing these values to those for the risk-neutral optimal plan, we see that the risk-seeking agent prefers a higher variance with a modest reduction in the expected total reward, and the risk-averse agent prefers a lower variance with a similar reduction.

However, a planning objective directly involving the variance is not an MEU objective in general. Otherwise, according to utility theory, the variances of non-deterministic outcomes would be the expectation of variances of deterministic outcomes, namely zero, which obviously is not true. The risk-sensitive objectives, however, provide a way to take the variance into account. Since a larger variance implies a better best-case scenario and a worse worst-case scenario, and conversely, a smaller variance implies a worse best-case scenario and a better worst-case scenario, the tradeoff between the mean and the variance is also a tradeoff among planning for the best-case, expected-case, and worst-case scenarios. For a risk-averse agent, the tradeoff is between the expected case and the worst case; for a risk-seeking agent, the tradeoff is between the expected case and the best case. In reality, the best-case and worst-case scenarios rarely happen, therefore planning based on such scenarios is often extremely biased. Even worse, the worst-case scenarios can have infinite loops, and thus are not suitable for planning problems with goal states, which will never be

[3] $\gamma = 0.8$.

reached. Risk-sensitive planning hence provides a well-founded alternative for agents that care about the variances.

1.5 Scope of Research

This thesis studies how to solve AI planning problems efficiently under uncertainty under risk-sensitive planning objectives. For this purpose, I formulate the planning problems as Markov decision process models with a finite number of states and actions, and consider indefinite planning horizons with goal states. I consider both exponential utility functions and general risk-sensitive utility functions in this research.

Most AI planning problems have a finite number of states and actions. AI planning problems are often described using a finite set of features, each with a finite number of values. This results in problems with a finite number of states and a finite number of actions.

AI planning problems often have goals to achieve. The most common type of goal is to reach a set of goal states. The agent will stop acting only when it is in a goal state. Planning horizons concern how far into the future the planner considers a plan. When goal states are present, it is appropriate to consider indefinite planning horizons that are characterized by reaching a goal state, but cannot be determined for certain in advance.

DT planning is closely related to reinforcement learning (Sutton and Barto, 1998; Kaelbling *et al.*, 1996). The major difference is that reinforcement learning assumes that the transition probabilities and the reward function are unknown, therefore a reinforcement learning method needs either to estimate the transition probabilities and the reward function (the model-based approach) or to estimate the optimal value function or an optimal policy directly (the model-free approach). On one hand, the model-based approaches essentially use a DT planning method to solve the problem after the model has been estimated. On the other hand, a model-free method can also be used to solve DT planning problems if the problem formulation, which has known transition probabilities and a known reward function, is used as a generative model. In this thesis, we consider only DT planning methods.

Exponential utility functions, as used in the example in Section 1.1, are a popular class of risk-sensitive utility functions (Corner and Corner, 1995). By varying a risk parameter, exponential utility functions can express a spectrum of constant risk attitudes such that the preferences among alternatives do not change with the wealth level. I consider planning with exponential utility functions, because they are the most widely used risk-sensitive utility functions, and also because they are of special structure and thus the planning methods can be customized to be more efficient.

I also consider more general risk-sensitive utility functions to model variable risk attitudes, where the preferences among alternatives can change with the wealth level. Human decision makers can be both risk-averse and risk-seeking at the same time, as exemplified by those buying lottery tickets and insurance at the same time. Such a risk attitude can only be modeled by general risk-sensitive utility functions representing variable risk attitudes, but not by exponential utility functions, since the latter can only model constant risk attitudes.

When discussing computational procedures for exponential and general risk-sensitive utility functions, I consider problems with only nonpositive rewards, since they can model action costs such as consumption of resources, which human decision makers care about. We also consider problems with only nonnegative rewards, since such problems have solution procedures almost identical to those with only nonpositive rewards.

It is often convenient to have arbitrary (both nonnegative and nonpositive) rewards in the formulation of AI planning problems. However, it is a subtle matter for risk-sensitive planning, since many basic properties such as the existence and finiteness of optimal values remain unknown. Therefore, for problems with arbitrary rewards, I discuss their basic properties in this thesis, which are prerequisites for solving such problems.

1.6 Contributions

As a preview of the contributions, Table 1.2 lists a rough classification of research effects for planning with MDPs under different planning objectives, and shows where this thesis

Table 1.2: Summary of existing research and contributions

Reward Model and Objective		Basic Properties	Basic Algorithms	Large-Scale Problems
MER$_\beta$		Well Known	Well Known	Active Research
Nonnegative/ Nonpositive Rewards	MER	Well Known	Well Known	Active Research
	MEU$_{\text{exp}}$	Recent	Recent	**Chapter 3**
	MEU	**Chapter 4**		—
Arbitrary Rewards	MER	Known	Known	—
	MEU$_{\text{exp}}$ and MEU	**Chapter 5**	—	—

work fits. In the table, MER$_\beta$ indicates the discounted risk-neutral planning objective and MEU$_{\text{exp}}$ indicates the risk-sensitive planning objective with an exponential utility function.

This work consists of three contributions related to risk-sensitive planning. For risk-sensitive planning objectives with exponential utility functions, I use a transformation-of-algorithms approach and show that many existing AI planners under a risk-neutral objective can be adapted to solve risk-sensitive planning problems. For risk-sensitive planning objectives with general risk-sensitive utility functions, I use a state-augmentation approach and construct basic dynamic programming algorithms to solve such risk-sensitive planning problems. For planning problems with arbitrary rewards, I investigate their basic properties, especially existence and finiteness properties under risk-sensitive planning objectives. In the following subsections, I discuss these contributions in more detail.

1.6.1 Exponential Utility Functions

The benefit of an exponential utility function is that a planning problem remains decomposable under the corresponding MEU planning objective. This makes it possible to reuse results for solving large problems under risk-neutral planning objectives.

I use transformations of algorithms that transforms a risk-neutral planner to an MEU$_{\text{exp}}$ planner. The transformations are of two principle types. The first type relates probabilities to pseudo-probabilities that may not sum to one. The second type relates (transition-dependent) discount factors to transition-dependent pseudo-discount factors that may be greater than one. These two types are almost equally applicable to MDPs whose transition probabilities are represented explicitly. But they differ when the probabilities are implicitly represented. Two kinds of implicit representations of probabilities are common in decision-theoretic planning: temporally extended probabilities that involve multiple time-steps, and probabilities represented in feature-based (or factored) forms. The key intuition is that the pseudo-probability transformation is more convenient for temporally extended probabilities, while the pseudo-discount factor transformation is more convenient for factored probabilities.

To demonstrate how the transformation-of-algorithms works, I use generalizations of existing methods for solving large-scale risk-neutral planning problems. The LAO* method, a search-based method (Hansen and Zilberstein, 2001), is used to show that if the probabilities are given explicitly, either type can lead to the correct algorithm. The pseudo-probability transformation is demonstrated using the sensor-planning method from (Hansen, 1994, 1997) and hierarchical decision-theoretic planning methods (Sutton *et al.*, 1999b; Dietterich, 2000; Parr, 1998). On the other hand, the pseudo-discount factor transformation is applied to structured dynamic programming methods for solving factored MDPs (Boutilier *et al.*, 2000; Hoey *et al.*, 1999; Feng and Hansen, 2002; Hansen *et al.*, 2002).

The transformation-of-algorithms approach has the advantage that ideas from solving large-scale planning problems under a risk-neutral objective can be reused to solve risk-sensitive planning problems, rather than developing completely new methods from scratch. When new methods for planning under a risk-neutral objective become available, they can be adapted to solve risk-sensitive planning problems. Since it is unlikely that there will be a universal planning method that can work for all kinds of AI planning problems under uncertainty, we will also need risk-sensitive planners that use different techniques, and the transformation approach is very attractive for its simplicity and ease of implementation to obtain risk-sensitive planners from their risk-neutral counterparts.

I emphasize that the transformation-of-algorithms is only nominal, in the sense that we can solve the problem using a method with visual resemblance to the corresponding method under a risk-neutral objective, but there might be conditions and separate proofs to ensure the correctness of the transformed planner, and thus I do not pursue an automatic algorithm transformer. The benefit of a nominal transformation is that it is easier to relate and communicate the underlying ideas, and can hint at the correct procedures when we consider whether an idea from planning under a risk-neutral objective can be reused to solve problems under risk-sensitive planning objectives. It requires only minimal changes to existing risk-neutral planners in order to obtain risk-sensitive planners.

1.6.2 General Risk-Sensitive Utility Functions

Different from the case of exponential utility functions, I consider basic computational procedures for planning with general risk-sensitive utility functions, as well as properties leading to the construction of such procedures.

For general risk-sensitive utility functions, I augment states with accumulated plan-execution rewards. This state-augmentation approach converts a planning problem under the MEU objective into a planning problem under the MER objective with an augmented state space, which in general has an infinite number of states. But the benefit is that we can use results for MDPs under the MER objective with infinite state spaces. An equivalence result is obtained for the original problems and the augmented problems, so that an optimal plan for the augmented problem is also an optimal plan for the original problem.

With the state-augmentation approach, we immediately obtain that if there exists an optimal plan, an optimal plan in general is not a function from states to actions. It also depends on how much reward has been accumulated starting from the initial state. In other words, there are optimal plans that are functions from the augmented states to actions.

The augmented model with augmented states in general has a countably infinite number of states since there are countably many possible accumulated rewards, so we need a method to deal with an infinite number of states. I use a functional interpretation of value functions and approximation techniques. More concretely, I use functional value functions that map (original) states to real-valued functions of the wealth level, and approximate each functional value function for each original state. The functional interpretation of value functions is possible since the augmented states that correspond to the same original states but different accumulated rewards share the same transition probabilities for any given action. We can then view that the (original) states have functional value functions that are mappings from (original) states to real-valued functions of the accumulated reward, while regular value functions are mappings from states to real numbers. However, to deal with the infinite state space, we still need to use some approximation method. The functional interpretation also allows for approximation techniques, for example, using piecewise linear functions so that approximately optimal values can be obtained efficiently. To fully understand the properties and complexities of this approximation technique, I first study finite horizon problems for which the approximation problem is simpler.

One-switch utility functions form an interesting class of utility functions (Bell, 1988; Bell and Fishburn, 2001). It has an attractive property that it models a class of smoothly varying risk attitudes, which cannot be modeled by exponential utility functions. It also has an attractive form as a linear combination of exponential and linear utility functions, two types of utility functions for which we know how to solve planning problems. This suggests that the planning problems with one-switch utility functions can be solved by combining results for linear, exponential, and general utility functions. We provide a value iteration procedure and a backward induction procedure for solving planning problems with one-switch utility functions.

1.6.3 Problems with Arbitrary Rewards

As mentioned in Section 1.5, the risk-sensitive planning methods, with either exponential or general risk-sensitive utility functions, are applicable to problems with pure action costs, which are the most common type of planning problems. Since costs are nonpositive rewards in the MDP formulation, MDP models of these planning problems are called negative in the MDP literature. Some of these methods are also applicable to positive models where the rewards are all nonnegative, since the same procedure often works for both positive and negative models, although for different reasons.

It can be more convenient for decision-theoretic planning to use arbitrary rewards that can be both nonpositive and nonnegative. One proposal is to use costs (negative rewards) associated with actions and arbitrary rewards associated with states (Boutilier *et al.*, 1999). However, using arbitrary rewards poses a problem for risk-sensitive planning, since many of the theoretical foundations are not yet known. The most important research question is to determine sufficient conditions under which the optimal expected utilities exist and are finite, conditions general enough to allow for most of the interesting AI planning problems.

I propose different sets of conditions, each of which consists of two classes of conditions: one that constrains the utility function and one that constrains the MDP model. For each class, we have a spectrum of conditions. These two classes of conditions are complementary in the sense that if we pick a stronger condition from one class, then we can pick a weaker condition from the other class, and the existence and finiteness properties should hold under this combination of conditions. In this way, we can characterize a wide range of problems that can be considered in the MDP framework.

In this thesis, I show that the existence and finiteness properties hold for a special class of plans, namely, stationary plans where the action to perform in each state does not change over time, under those different sets of conditions. We conjecture that the existence and finiteness properties also hold for all possible plans.

1.6.4 Summary

I argue that the current decision-theoretic planners have shortcomings that limit their applications, since they lack the ability to deal with risk attitudes that arise in planning under uncertainty. I therefore combine ideas from AI planning, operations research, and utility theory to study the basic properties and computational issues involved. The fusion of these methods is most promising for risk-sensitive planning. Utility theory provides the fundamental theoretical background for the MEU approach to planning. But utility theory does not answer the question of how to construct complex plans. Operations research then provides basic computational approaches in the risk-neutral case, which is the starting point of this research. Operations research also provides some tools for obtaining theoretical results for risk-sensitive planning, such as the existence of optimal expected values and plans, and the structures of optimal or near-optimal plans. But operations research methods for solving MDPs often are not sufficient for large-scale problems under general MEU objectives. AI planning, on the other hand, focuses on using knowledge about the structure of problems to solve large-scale problems and speed up the solution process. I show that we then can use ideas from various AI planning methods to solve planning problems under risk-sensitive planning objectives. To my knowledge, this work is the first comprehensive investigation of general risk-sensitive planning objectives in AI planning.

CHAPTER II

BACKGROUND

This chapter provides the background information on which this work is built, and reviews related work from the literature. In Section 2.1, we review concepts and results from utility theory, including the principle of maximization of expected utility and risk-sensitive utility functions. Section 2.2 formally defines Markov decision process models and related concepts. Section 2.2 also introduces risk-sensitive planning objectives and a series of related problems for solving MDPs under risk-sensitive planning objectives. In Section 2.3, we review basic results from the literature for solving MDPs under risk-neutral planning objectives. These results are important since risk-neutral planning objectives can be considered a special case of risk-sensitive planning objectives, and on the other hand, our treatment of risk-sensitive planning objectives is built on results for risk-neutral planning objectives. Section 2.4 briefly reviews results for risk-sensitive planning objectives from both the artificial intelligence planning and operations research literatures. Section 2.5 reviews recent developments in solving large-scale MDPs under risk-neutral objectives, whose ideas are reused as the basis for solving large-scale MDPs under risk-sensitive objectives.

2.1 Utility Theory

Utility theory provides a formal foundation for expressing risk-sensitive planning objectives. Utility theory is a normative theory for decision making (von Neumann and Morgenstern, 1944; Fishburn, 1970; Barberá *et al.*, 1998). As a component of decision theory, it focuses on formal properties of preference structures. For this thesis, we adopt the version of (von Neumann and Morgenstern, 1944), also known as the von Neumann-Morgenstern (vNM) utility theory.

2.1.1 Principle of Maximization of Expected Utility

Utility theory studies preference structures and their numerical representations in form of utility functions. In this section, we introduce the preference relations and results about the existence of utility functions. We also discuss how decisions are made with the aid of utility functions.

A preference structure can be formulated as a binary preference relation. Suppose $\mathcal{Z}$ is a set of outcomes resulting from an agent's decisions. We assume that the agent's preference structure over $\mathcal{Z}$ is captured by a preference relation $\succ$ on $\mathcal{Z}$, defined as a binary relation with the following property: for all outcomes $A, B \in \mathcal{Z}$,

- A is preferred over B if and only if $A \succ B$.

It is also convenient to define two more relations $\sim$ and $\succsim$ as follows: for all outcomes $A, B \in \mathcal{Z}$,

- A is indifferent to B if and only if $A \sim B$, that is, neither $A \succ B$ nor $B \succ A$, and
- $A \succsim B$ if and only if $A \succ B$ or $A \sim B$.

The preference relation $\succsim$ is assumed to be a total order, that is, every pair of outcomes are comparable. Formally, the relation $\succsim$, as induced by $\succ$ and thus $\sim$, is a binary relation with the following three properties:

Reflexivity: For all outcomes $A \in \mathcal{Z}$, $A \sim A$.

Orderability: For all outcomes $A, B \in \mathcal{Z}$, exactly one of the following holds: $A \succ B$, $B \succ A$, or $A \sim B$.

Transitivity: For all outcomes $A, B, C \in \mathcal{Z}$, if $A \succsim B$ and $B \succsim C$, then $A \succsim C$.

Preference structures can be extended to uncertain outcomes. Uncertain outcomes are represented using (discrete) distributions over $\mathcal{Z}$, referred to as lotteries. A lottery L with a finite number of distinct outcomes $C_1, C_2, \ldots, C_n \in \mathcal{Z}$ and respective probabilities $p_1, p_2, \ldots, p_n$ is written as

$$L = [p_1, C_1; p_2, C_2; \ldots ; p_n, C_n], \qquad \text{where} \qquad \sum_{i=1}^{n} p_i = 1, \quad p_i \geq 0, \ i = 1, \ldots, n.$$

A lottery is degenerate if $n = 1$, in which case we omit the probability in the notation. A lottery can also have a countably infinite number of distinct outcomes $C_1, C_2, \ldots, C_n, \ldots \in \mathcal{Z}$ with respective probabilities $p_1, p_2, \ldots, p_n, \ldots$, which can be written as

$$L = [p_1, C_1; p_2, C_2; \ldots; p_n, C_n; \ldots], \qquad \text{where} \qquad \sum_{i=1}^{\infty} p_i = 1, \quad p_i > 0,\ i = 1, 2, \ldots$$

We denote as $L(\mathcal{Z})$ the set of all lotteries over the outcome set $\mathcal{Z}$. The agent's preference relation is correspondingly extended to lotteries, and we still use the notation $\succ, \sim, \succsim$ for the preference relation on lotteries. Preference structures over lotteries are also assumed to have additional properties that are consistent with human intuition about preference structures involving uncertainty (following Russell and Novig, 2002, Chapter 16):

Continuity: For all lotteries $A, B, C \in L(\mathcal{Z})$, if $A \succ B \succ C$, then there exists $p \in (0, 1)$ such that

$$[p, A; 1 - p, C] \sim B.$$

Substitutability: For all lotteries $A, B \in L(\mathcal{Z})$, if $A \sim B$, then for all lotteries $C \in L(\mathcal{Z})$ and all $p \in [0, 1]$,

$$[p, A; 1 - p, C] \sim [p, B; 1 - p, C].$$

Monotonicity: For all lotteries $A, B \in L(\mathcal{Z})$, if $A \succ B$, then for all $p, q \in [0, 1]$, $p \geq q$ if and only if

$$[p, A; 1 - p, B] \succsim [q, A; 1 - q, B].$$

Decomposability: For all lotteries $A, B, C \in L(\mathcal{Z})$ and all $p, q \in [0, 1]$,

$$\big[p, A; 1 - p, [q, B; 1 - q, C]\big] \sim [p, A; (1 - p)q, B; (1 - p)(1 - q), C].$$

These properties imply that the extended preference relation is also a total order on $L(\mathcal{Z})$.

A fundamental result from utility theory is the existence of utility functions (Fishburn, 1970). If the extended preference relation satisfies the above set of seven properties, also referred to as the utility axioms, then there exists a utility function $U : L(\mathcal{Z}) \mapsto \mathbb{R}$ such that for all lotteries $A, B \in L(\mathcal{Z})$, $A \succsim B$ if and only if $U(A) \geq U(B)$, where the utility of a lottery L is the expectation of the utilities of its components:

$$U(L) = U\big([p_1, C_1; p_2, C_2; \ldots; p_n, C_n; \ldots]\big) = \sum_{i=1}^{\infty} p_i \cdot U(C_i) = E\big[U(\mathbf{z})\big], \tag{2.1}$$

where $\mathbf{z}$ is the random variable for the outcomes of lottery L. Therefore, the utility function only needs to be defined for elements in $\mathcal{Z}$, and its definition on $L(\mathcal{Z})$ can be obtained using Eq. (2.1). Moreover, the utility function is unique up to a positive linear transformation, that is, both $U_1(\cdot)$ and $U_2(\cdot) = aU_1(\cdot) + b$ (where $a > 0$) are utility functions corresponding to the same preference structure. In applications, it is not hard to determine a normalized utility function whose a and b values are uniquely determined, if we assign arbitrary distinct values (0 and 1, for example) to a pair of distinct outcomes that are not indifferent. For example, the concave exponential utility function used in Section 1.1 is normalized so that a zero wealth level corresponds to a utility of zero and a positive infinity wealth level corresponds to a utility of one.[1] Without loss of generality, in this thesis, we assume that the utility function is always normalized in such a way.

Consequently, a rational agent facing a set of finite lotteries $\mathcal{L} \subseteq L(\mathcal{Z})$ will choose the lottery that is maximal in $\mathcal{L}$, that is, the lottery that maximizes the expected utility of all possible outcomes. This is the principle of maximization of expected utility (MEU). Under this principle, the decision making problem can be solved with the aid of utility functions. The agent needs to find the maximum expected utility

$$U^* = \max_{L \in \mathcal{L}} U(L).$$

If this is possible, we will make an optimal decision by choosing

$$L^* \in \arg\max_{L \in \mathcal{L}} U(L).$$

However, if there are an infinite number of lotteries, we may only be able to make approximately optimal decisions. In this case, it is possible that the maximal utility does

[1] Concave exponential utility functions are special in that the utility of positive infinity is well-defined, which is not the case in general.

not exist, but we can always obtain the supremum of the expected utility

$$U^* = \sup_{L \in \mathcal{L}} U(L).$$

If the supremum cannot be reached by choosing a lottery from $\mathcal{L}$, we cannot obtain an optimal decision. However, if U^* is finite, we can approximate the optimal utility U^* to within an arbitrary error. There are two cases where the optimal utility is finite:

- As discussed in (Fishburn, 1967), the original formulation of the vNM-utility theory leads to bounded utility functions, that is, $U(\cdot)$ is finite for all outcomes (and thus for all lotteries). In this case, U^* is finite.

- However, it is sometimes convenient to allow for unbounded utility functions (Fishburn, 1975), by using technically more subtle sets of utility axioms. In this case, Kennan (1981) provided a set of sufficient conditions on the utility function so that the MEU principle can still be used. The essential one from his conditions is that the optimal utility U^* is finite. Notice that in this case, the utility function (of lotteries) is bounded from above, but can still be unbounded from below.

In either case, we will then seek an ϵ-optimal decision L^*_ϵ such that for a given error $\epsilon > 0$,

$$U(L^*_\epsilon) \geq U^* - \epsilon.$$

ϵ-optimal decisions are meaningful since vNM-utility theory defines a cardinal utility, that is, the utilities of different lotteries not only indicate the ordering of these lotteries, but also the strength of preferences among the lotteries (Köbberling, 2002). For lotteries $A, B, C, D \in L(\mathcal{Z})$, suppose that $U(A) > U(B)$ and $U(C) > U(D)$. If $U(A) - U(B) > U(C) - U(D)$, then the strength of the preference of A over B is greater than the strength of the preference of C over D. This property allows us to find an ϵ-optimal decision if an optimal decision does not exist. Since we assume that the utility function is normalized, the desired approximation error can be often selected based on the specific form of the utility function. We also seek an ϵ-optimal decision if an optimal decision is hard to obtain. This is especially important for decision-theoretic planning, where we often want to obtain a "good" plan fast, rather than spending a lot of extra computational effort on finding a marginally better optimal one.

On the other hand, according to utility theory, any real-valued function on $\mathcal{Z}$ defines a preference structure through Eq. (2.1). In fact, the risk-sensitive utility functions that we discuss next can be seen as defining risk-sensitive preference structures in this way. However, it would be more convenient to define preference structures based on real-valued functions on $L(\mathcal{Z})$ directly for decision problems under uncertainty. According to utility theory, a real-valued function on $L(\mathcal{Z})$ can define a preference structure satisfying the utility axioms if and only if it satisfies the functional form of the utility axioms, namely

Continuity: For all lotteries $A, B, C \in L(\mathcal{Z})$, if $U(A) > U(B) > U(C)$, then there exists $p \in (0, 1)$ such that

$$U\big([p, A; 1-p, C]\big) = U(B).$$

Substitutability: For all lotteries $A, B \in L(\mathcal{Z})$, if $U(A) = U(B)$, then for all lotteries $C \in L(\mathcal{Z})$ and all $p \in [0, 1]$,

$$U\big([p, A; 1-p, C]\big) = U\big([p, B; 1-p, C]\big).$$

Monotonicity: For all lotteries $A, B \in L(\mathcal{Z})$, if $A \succ B$, then for all $p, q \in [0, 1]$, $p \geq q$ if and only if

$$U\big([p, A; 1-p, B]\big) \geq U\big([q, A; 1-q, B]\big).$$

Decomposability: For all lotteries $A, B, C \in L(\mathcal{Z})$ and all $p, q \in [0, 1]$,

$$U\Big(\big[p, A; 1-p, [q, B; 1-q, C]\big]\Big) = U\Big(\big[p, A; (1-p)q, B; (1-p)(1-q), C\big]\Big).$$

Since the relation $>$ is a total order on $\mathbb{R}$, we do not need to include the first three utility axioms: reflexibility, orderability, and transitivity.

2.1.2 Utility Functions on the Real Line and Risk Attitudes

Since planning is based on the rewards gathered in the process of execution, it is convenient to define the relevant utility functions on $\mathcal{Z} = \mathbb{R}$. In this case, the argument of a utility function is usually referred to as wealth level, denoted by w. For such a utility function, it is intuitive to require that the utility function be monotonically nondecreasing in w. Sometimes, we can further assume that the utility function is continuous in w.

For a utility function that is strictly monotonically increasing and continuous in w, the certainty equivalent of a lottery $L = \mathbf{w}$ is

$$\mathrm{ce}_U(\mathbf{w}) = U^{-1}\Big(E\big[U(\mathbf{w})\big]\Big),$$

where we use $\mathbf{w}$ to emphasize that the random outcome of L is a wealth level. If an agent is faced with a nondegenerate lottery, then the agent is indifferent between participating in the lottery and getting a deterministic reward whose amount equals the certainty equivalent. This also holds for degenerate lotteries trivially.

Agents can have different risk attitudes, that is, different ways of evaluating a (nondegenerate) lottery. Risk attitudes are modeled using the agents' utility functions. Based on the utility functions, we can identify different types of risk attitudes, which can be traced back to (Friedman and Savage, 1948):

- if $U\big(E[\mathbf{w}]\big) = E\big[U(\mathbf{w})\big]$ for all nondegenerate lotteries $\mathbf{w}$, then the utility function and thus the agent are risk-neutral;
- if $U\big(E[\mathbf{w}]\big) > E\big[U(\mathbf{w})\big]$ for all nondegenerate lotteries $\mathbf{w}$, then the utility function and thus the agent are risk-averse; and
- if $U\big(E[\mathbf{w}]\big) < E\big[U(\mathbf{w})\big]$ for all nondegenerate lotteries $\mathbf{w}$, then the utility function and thus the agent are risk-seeking.

A risk-neutral utility function is linear, and thus is equivalent to an identity function; a utility function is risk-sensitive if and only if it is not risk-neutral, that is, if and only if it is nonlinear. Moreover, an agent is risk-averse if and only if its utility function is strictly concave, and an agent is risk-seeking if and only if its utility function is strictly convex. Risk aversion and risk seekingness defined above are global properties of risk attitudes, since these properties do not change with the wealth level.

Pratt (1964) defined local properties of risk attitudes as well as a measure of risk-sensitivity, which are convenient for risk attitudes involving both risk aversion and risk seekingness, such as the risk attitude of a person who buys insurance (risk-averse) and lottery tickets (risk-seeking) at the same time. The local risk property is related to the local convexity of the utility function. If the utility function is twice differentiable and strictly monotonically increasing, Pratt (1964) defined the risk measure as

$$R_U(w) = -\frac{U''(w)}{U'(w)}. \tag{2.2}$$

The local risk measure $R_U(w)$ relates to the global risk properties as follows: if $R_U(w) = 0$ everywhere, then the utility function is linear, and thus the agent is risk-neutral; if $R_U(w) > 0$ everywhere, then the utility function is strictly concave, and thus the agent is risk-averse; and if $R_U(w) < 0$ everywhere, then the utility function is strictly convex, and thus the agent is risk-seeking.

One reason why the risk measure takes this particular form rather than just the second derivative, which also indicates local convexity, is that this definition is invariant with respect to a positively linear transformation of utility functions, which does not change the agent's preference structure, and thus does not change its risk attitude. In other words, a risk measure function uniquely determines a utility function that is twice differentiable and strictly monotonically increasing (up to a positively linear transformation), and vice versa.

Another reason why the risk measure takes the form in Eq. (2.2) is that the risk measure $R_U(w)$ also provides an interpretation for nonlinear utility functions as a way of making tradeoffs between the mean and the variance of a lottery. Suppose that $\mathbf{w} \in L(\mathbb{R})$ is a lottery on the real line. Then a direct tradeoff between the mean $E[\mathbf{w}]$ and the variance $\mathrm{Var}[\mathbf{z}]$ is in the form of $k_1 E[\mathbf{w}] + k_2 \mathrm{Var}[\mathbf{w}]$ where $k_1, k_2 \in \mathbb{R}$. However, this is not a utility function in the vNM-utility theory, since it violates the substitutability property, for example. This violation can be verifies as follows. It is easy to see that the mean $E[\mathbf{w}]$ is a well-defined utility function for $\mathbf{w}$. Therefore the violation can be verified by checking the variance $\mathrm{Var}[\mathbf{w}]$ alone. Consider (deterministic) outcomes $A, B, C \in \mathbb{R}$, where $A = 0, B = 10$, and $C = 1$. Then $\mathrm{Var}[A] = \mathrm{Var}[B] = \mathrm{Var}[C] = 0$. Now consider $D = [0.5, A; 0.5, C]$ and

$E = [0.5, B; 0.5, C]$. If the substitutability property held for the variance, we should have $\mathrm{Var}[D] = \mathrm{Var}[E]$, but in fact $\mathrm{Var}[D] = 0.25 \neq \mathrm{Var}[E] = 20.25$. Actually, we can also verify that the variance violates the monotonicity property as well.

However, it is intuitively appealing to have a planning objective related to variances. Risk-sensitive utility functions can play such a role. In fact, if the variance is small, the expected risk-sensitive utility can be viewed as an approximate tradeoff of the mean and the variance. Suppose that the agent is faced with a lottery $\mathbf{w}$ with a small variance. Let $w_0 = E[\mathbf{w}]$. Pratt (1964) showed that the risk measure and the certainty equivalent of $\mathbf{w}$ have the following relation

$$\mathrm{ce}_U(\mathbf{w}) \approx w_0 - \frac{1}{2}\mathrm{Var}[\mathbf{w}_0] \cdot R_U(w_0) = E[\mathbf{w}] - \frac{1}{2}\mathrm{Var}[\mathbf{w}] \cdot R_U(E[\mathbf{w}]).$$

Therefore, if the agent is risk-averse, that is, $R_U(w) > 0$, then the certainty equivalent of $\mathbf{w}$ is lower than the expected outcome by an amount that is approximately proportional to the variance, and thus prefers a lottery with a smaller variance. Likewise, if the agent is risk-seeking, that is, $R_U(w) < 0$, it prefers a lottery with a larger variance. This indicates that utility theory can be used to make tradeoffs among the worst-case, expected-case, and best-case outcomes, where the worst-case scenario suffers from a large variance, and the best-case scenario benefits from a large variance.

2.2 Markov Decision Process Models

We are interested in AI planning problems under uncertainty. Many of these problems can be modeled as Markov decision process (MDP) problems. In such a problem, the agent has opportunities to make decisions to influence its state in the environment. The influence is nondeterministic in nature, and is modeled probabilistically. The agent needs to plan a set of actions according to a given planning objective.

2.2.1 Definitions

The agent can make decisions at a set of time points, known as decision epochs. We consider discrete decision epochs, denoted by natural numbers $t \in \mathbb{N}$. A Markov decision process (MDP) model M is a 4-tuple (S, A, P, r) consisting of

- a state space S;
- an action space A, where the set of available actions in state s is A_s, and $A = \bigcup_{s \in S} A_s$;
- a transition probability distribution function $P : S \times A \rightharpoonup \mathcal{P}(S)$, where $\mathcal{P}(S)$ is the space of probability distributions over the state space S, and $P(s'|s,a)$ denotes the probability of resulting in state $s' \in S$ starting from state $s \in S$ and executing $a \in A_s \subseteq A$; and
- a reward function $r : S \times A \times S \rightharpoonup \mathbb{R}$, where $r(s,a,s')$ is the immediate reward received for the transition (s,a,s') where $P(s'|s,a) > 0$.

Notice that the transition probability P is a partial function that is only defined for all states $s \in S$ and all actions $a \in A_s$. The reward function is also a partial function that is not defined for all combinations of (s,a,s') where $s, s' \in S$ and $a \in A$, but only for those where the transition probability $P(s'|s,a)$ is defined and positive. We refer to such transitions as valid transitions.

We are especially interested in finite MDPs, that is, MDPs whose state and action spaces are finite. Our future discussions assume that the state and action spaces are finite unless we explicitly state otherwise.

Condition 2.1 (Finite Model)**.** The state space S and the action space A are finite.

For later reference (especially for general utility functions in Chapter 4), we also need results for models with countably infinite states. But we restrict the number of available actions in each state to be finite.

Condition 2.2 (Countable States)**.** The state space S is countable, and the set of available actions A_s is finite for each $s \in S$.

Condition 2.1 implies Condition 2.2. The reason is that if S and A are finite, then $A_s \subseteq A$ is also finite is also finite. In fact, only under Condition 2.2, the above definitions of the transition probability distribution and the reward function are meaningful. Otherwise, additional technical conditions are needed to make the model well-defined (Feinberg and Shwartz, 2002).

The planning horizon (or horizon for short) is the number of decision epochs that the planner considers, and it can be finite or infinite. In the finite horizon case, we refer to the last decision epoch as T where $T \geq 1$ and no decision is made at or after $t = T$, corresponding to the termination of the MDP. We also allow $T = 0$ for convience. In the infinite horizon case, the agent acts forever unless a goal state $g \in G$ is reached (to be discussed shortly).

For later convenience, we define the successor set of the pair of a state $s \in S$ and an action $a \in A_s$ to be

$$\mathrm{succ}(s, a) = \big\{ s' \,\big|\, s' \in S, P(s'|s, a) > 0 \big\}$$

and the successor set of a state $s \in S$ is

$$\mathrm{succ}(s) = \bigcup_{a \in A_s} \mathrm{succ}(s, a).$$

The predecessor set of the pair of a state $s \in S$ and an action $a \in A$ is defined as

$$s' \in \mathrm{pred}(s, a) \qquad \text{if and only if} \qquad s \in \mathrm{succ}(s', a),$$

and the predecessor set of a state $s \in S$ is defined as

$$s' \in \mathrm{pred}(s) \qquad \text{if and only if} \qquad s \in \mathrm{succ}(s').$$

An MDP problem consists of an MDP model, a set of initial states, a set of goal states, and a planning objective. The set of initial states is $S_0 \subseteq S$ where $S_0 \neq \varnothing$, since the agent has to start acting from some state. Often, the set of initial states S_0 is not explicitly given, in which case, we mean $S_0 = S$ and all states can be an initial state. The set of goal states are denoted as $G \subseteq S$. Reaching a goal state indicates the termination of the agent's actions. In this sense, MDP problems with non-empty goal states are problems with indefinite planning horizons. Equivalently, we may also use an implicit representation of goal states, by assuming that only a special action a_{null} is allowed in a goal state g, such that $P(g|g, a_{\text{null}}) = 1$, and $r(g, a_{\text{null}}, g) = 0$. In this way, we can treat goal states in the same way as ordinary states, which is sometimes convenient mathematically. It is possible that $G = \varnothing$, which means that the agent will never stop acting unless the planning horizon is reached (in the finite horizon case). In this thesis, we mainly discuss MDP problems with goal states, and use both the explicit and implicit representation of goal states, depending on which one is more convenient. We defer the discussion of planning objectives until Section 2.2.2.

The agent starts acting in an initial state $s_0 \in S_0$. At decision epoch t, the agent is in state s_t. The agent performs an action $a_t \in A_{s_t}$, then transitions into a new state s_{t+1} according to the transition probability distribution $P(s_{t+1}|s_t, a_t)$, and receives an immediate reward $r_t = r(s_t, a_t, s_{t+1})$. Therefore, for the transition (s, a, s'), we can refer to s as the current-time state, and s' as the next-time state. The qualifier Markov is used since the transition probability distributions only depend on the previous state and the action performed, which is known as the Markovian property of transition probabilities.

A history at decision epoch t is the sequence of states and actions performed, from the initial state through the current state. Formally, a history is $h_t = (s_0, a_0, \dots, s_{t-1}, a_{t-1}, s_t)$, which follows the recursive relation $h_0 = (s_0)$ and $h_t = h_{t-1} \circ (a_{t-1}, s_t)$. We define the set of all histories recursively as

$$H_0 = S, \qquad H_{t+1} = \{h_t \circ (a_t, s_{t+1}) \,|\, h_t \in H_t, a_t \in A_{s_t}, P(s_{t+1}|s_t, a_t) > 0\}. \tag{2.3}$$

This definition differs from a common definition of history, where $H_t = (S \times A)^t \times S$, since many elements in the common definition of H_t are not realizable. Our definition is preferred to simplify proofs in Chapter 4. A similar definition has been used in (Hinderer, 1970). Let $H = \bigcup_{t=0}^{T} H_t$ or $H = \bigcup_{t=0}^{\infty} H_t$ be the set of all histories of finite or infinite horizons, respectively. A trajectory h is a "complete" history whose length is $2T + 1$ (for a finite horizon problem) or countably infinite (for an infinite horizon problem). The set of trajectories are H_T and H_∞, respectively. With a slight abuse of notation, we also use h to denote a history with an unspecified length. The exact meaning should be clear from the context. In the latter case, we also use h_t to denote the prefix of h up to decision epoch t.

A decision rule is used by the agent to determine which action to perform in the current state. A deterministic history-dependent (HD) decision rule at decision epoch t is a mapping $d_t : H_t \mapsto A_{s_t}$. A randomized history-dependent (HR) decision rule at decision epoch t is a

$$\begin{tikzcd}
 & & \Pi^{\mathrm{HR}} = \Pi \arrow[dl] \arrow[dr] & \\
 & \Pi^{\mathrm{MR}} \arrow[dl] \arrow[dr] & & \Pi^{\mathrm{HD}} \arrow[dl] \\
\Pi^{\mathrm{SR}} \arrow[dr] & & \Pi^{\mathrm{MD}} \arrow[dl] & \\
 & \Pi^{\mathrm{SD}} & &
\end{tikzcd}$$

Figure 2.1: The relationships among different classes of policies

mapping $d_t : H_t \mapsto \mathcal{P}(A_{s_t})$, where $\mathcal{P}(A_{s_t})$ indicates a probability distribution over the action set A_{s_t}. The probability of performing action $a \in A_{s_t}$ under an HR decision rule d_t thus is denoted by $d_t(h_t, a)$. Markovian decision rules are an important class of decision rules, where the dependence upon the history is only through the current state s_t. A deterministic Markovian (MD) decision rule at decision epoch t is a mapping $d_t : S \mapsto A_{s_t}$. A randomized Markovian (MR) decision rule at decision epoch t is a mapping $d_t : S \mapsto \mathcal{P}(A_{s_t})$. The probability of applying action $a \in A_{s_t}$ under an MR decision rule d_t is denoted by $d_t(s_t, a)$. We denote the class of a particular type of decision rules at decision epoch t by D_t^K, where $K \in \{\mathrm{HR}, \mathrm{HD}, \mathrm{MR}, \mathrm{MD}\}$.

In the MDP literature, a plan is traditionally referred to as a policy.[2] A policy, denoted as π, is a sequence of decision rules corresponding to the decision epochs. If the horizon is finite, a policy is a sequence of T decision rules, $\pi = (d_0, d_1, \ldots, d_{T-1})$; otherwise, a policy is an infinite sequence of decision rules, $\pi = (d_0, d_1, \ldots, d_t, \ldots)$. We also denote the different types of policies according to the types of their component decision rules by Π^K, where $K \in \{\mathrm{HR}, \mathrm{HD}, \mathrm{MR}, \mathrm{MD}\}$. The set of policies $\Pi^K = D_0^K \times D_1^K \times \ldots \times D_{T-1}^K$ if the planning horizon is finite, and $\Pi^K = D_0^K \times D_1^K \times \ldots$ if the planning horizon is infinite. Moreover, a policy is stationary if it is Markovian and $d_t = d$ for all t. If the policy π is stationary, $\pi = (d, d, \ldots)$, we use the notation $\pi(s, a) = d(s, a)$ if π is randomized and $\pi(s) = d(s)$ if π is deterministic. We denote the class of deterministic stationary (SD) policies by Π^{SD} and the class of randomized stationary (SR) policies by Π^{SR}. The class of all possible policies

[2] There are some differences in the literature in addition to the traditions of different communities, but we use them as synonyms in this thesis.

Π is just Π^{HR}. The relationships among the different classes of policies are illustrated in Figure 2.1, where the arrows lead from more general cases to more specific classes.

For a given policy π, we let P^{π} denote probabilities under this policy. We also use $P^{s,\pi}$ as a shorthand to indicate a probability under π where the initial state $s_0 = s$.

Under an HR policy $\pi = (d_0, d_1, \ldots, d_t, \ldots) \in \Pi^{\mathrm{HR}}$, the probability of realizing a history $h_t = (s_0, a_0, s_1, a_1, \ldots, s_t)$ is

$$\begin{aligned} &P^{s_0,\pi}(h_t) = P^{s_0,\pi}(s_0, a_0, s_1, a_1, \ldots, s_t) \\ &= d_0(s_0, a_0)P(s_1|s_0, a_0)d_1(h_1, a_1)P(s_2|s_1, a_1) \cdots d_{t-1}(h_{t-1}, a_{t-1})P(s_t|s_{t-1}, a_{t-1}). \end{aligned} \quad (2.4)$$

Under an MR policy $\pi = (d_0, d_1, \ldots, d_t, \ldots) \in \Pi^{\mathrm{MR}}$, this probability is simplified to

$$\begin{aligned} &P^{s_0,\pi}(h_t) = P^{s_0,\pi}(s_0, a_0, s_1, a_1, \ldots, s_t) \\ &= d_0(s_0, a_0)P(s_1|s_0, a_0)d_1(s_1, a_1)P(s_2|s_1, a_1) \cdots d_{t-1}(s_{t-1}, a_{t-1})P(s_t|s_{t-1}, a_{t-1}). \end{aligned}$$

Under an HD or MD policy π, this probability is further simplified to

$$P^{s_0,\pi}(h_t) = P^{s_0,\pi}(s_0, a_0, s_1, a_1, \ldots, s_t) = P_0(s_1|s_0, a_0)P_1(s_2|s_1, a_1) \cdots P_{t-1}(s_t|s_{t-1}, a_{t-1}),$$

where $a_t = d_t(h_t)$ or $a_t = d_t(s_t)$, respectively. Under an SR policy π, the MDP model is reduced to a homogeneous Markov chain (Kemeny and Snell, 1960; Kemeny *et al.*, 1976). If $\pi \in \Pi^{\mathrm{SR}}$, the transition probabilities are

$$P^{\pi}(s'|s) = \sum_{a \in A_s} \pi(s, a)P(s'|s, a).$$

If $\pi \in \Pi^{\mathrm{SD}}$, the transition probabilities are

$$P^{\pi}(s'|s) = P(s'|s, \pi(s)).$$

2.2.2 Planning Objectives

The MDP problem is to find the “best” policy for the given MDP model. However, there can be different ways of defining whether a policy is better than another one. So there are possibly different “best” policies under different planning objectives (also referred to as optimality criteria in the operations research literature). For a given initial state, a policy determines the possible trajectories and their respective probabilities according to Eq. (2.4). In the terminology of utility theory, for MDP problems, the set of outcomes is the set of all possible trajectories, and the set of lotteries is the set of all possible policies. Therefore, we can compare the policies based on the expected utility of the trajectory. As we discussed in Section 2.1.1, once we define the utility function of a trajectory, we have also defined the expected utility for a policy according to Eq. (2.1) and Eq. (2.4). We will compare policies using their expected utilities following utility theory. In other words, our planning objective is to maximize the expected risk-sensitive utility of the trajectory.

2.2.2.1 Finite Horizon

In some AI planning problems, there is a predetermined finite planning horizon, either suggested by the problem itself or set as a parameter when invoking the planner. The agent stops acting when the planning horizon is reached.

For finite horizon problems, we use the subscript T to indicate the finite planning horizon. We assume that the utility of a trajectory is fully captured by a utility function of the reward sequence of the trajectory, that is,

$$\vec{U}(h) = \vec{U}(s_0, a_0, s_1, a_1, \ldots, s_{T-1}, a_{T-1}, s_T) = \vec{U}(r_0, r_1, \ldots, r_{T-1}),$$

where the notation $\vec{U}$ emphasizes that the utility function is multi-dimensional, and $r_0, r_1, \ldots, r_{T-1}$ are the immediate rewards received at each decision epoch of the trajectory such that $r_t = r(s_t, a_t, s_{t+1})$.

In this thesis, we are especially interested in risk-sensitive (and risk-neutral) utility functions of accumulated rewards. In this case, we assume that the utility function can be further decomposed into two parts, where one part summarizes a sequence of immediate rewards into a single accumulated reward, and the other part represents the risk attitude toward the accumulated reward, which is a random variable. In the finite horizon case, the decomposition can be represented as

$$\vec{U}(h) = \vec{U}(r_0, r_1, \ldots, r_{T-1}) = U_{\text{risk}}\big(U_{\text{sum},T}(r_0, r_1, \ldots, r_{T-1})\big),$$

where U_{risk} is a one-dimensional utility function, and $U_{\text{sum},T}$ is a summarization utility function summarizing the immediate reward sequences over a finite horizon $T \geq 1$. The total (undiscounted) reward is a summarization function defined as

$$U_{\text{sum},T}(r_0, r_1, \ldots, r_{T-1}) = w_T = \sum_{t=0}^{T-1} r_t.$$

For completeness, we define $w_0 = 0$. Therefore, the risk-sensitive (and risk-neutral) utility of h is defined as

$$U(h) = \vec{U}(h) = U\left(\sum_{t=0}^{T-1} r_t\right) = U(w_T),$$

where for simplicity, we use U with a slight abuse of notation to indicate both the utility function of a history and the one-dimensional utility function U_{risk}.

Consequently, the expected utility of the state s for the policy π under a risk-sensitive planning objective[3] with the utility function U over a finite horizon T, is defined as

$$v_{U,T}^{\pi}(s) = E^{s,\pi}\big[U(h)\big] = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] = E^{s,\pi}\big[U(w_T)\big]. \tag{2.5}$$

The expected utility is state-specific since the probability distribution of the trajectory induced by a given policy only makes sense if the trajectories have the same initial state s_0. Notice that we use the superscripts of the expectation to indicate the initial state and the given policy. In this thesis, we use the convention that superscripts of the expectation symbol indicate known information, and subscripts indicate random variables. These scripts may be omitted if it is unnecessary to make the distinction.

The MEU principle then requires us to obtain (or approximate) the optimal expected utility of the state s under a risk-sensitive planning objective with the utility function U over a finite horizon T, defined as

$$v_{U,T}^{*}(s) = \sup_{\pi \in \Pi} E^{s,\pi}\big[U(h)\big] = \sup_{\pi \in \Pi} E^{s,\pi}\big[U(w_T)\big],$$

for the given initial state s.

Under Condition 2.2 (Countable States), for all initial states $s_0 = s \in S$ and all policies $\pi \in \Pi$, the expected utility $v_{U,T}^{\pi}(s)$ exists and is bounded, since the number of possible

[3] We consider the risk-neutral planning objective as a special case of the risk-sensitive planning objective where a linear (or equivalently, identity) utility function is used.

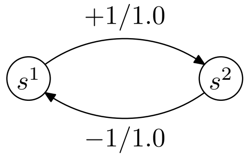


Figure 2.2: An example for the (non)-existence of values

trajectories is finite for finite horizon problems, and thus there are only a finite number of outcomes at each decision epoch and there are a finite number of decision epochs. Consequently, the optimal expected utility $v^*_{U,T}(s)$ exists and is finite.

2.2.2.2 Infinite Horizon

Typically, we need to deal with an infinite planning horizon, since there is no predefined finite planning horizon in many problems. The expected utility over an infinite horizon is defined as the limit of a series of finite-horizon expected utilities as the horizon approaches infinity. Formally, we denote the expected utility of the state s for the policy π under a risk-sensitive utility function with the utility function as

$$v^\pi_U(s) = \lim_{T\to\infty} v^\pi_{U,T}(s). \tag{2.6}$$

The expected value exists if and only if the limit converges on the extended real line $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$, namely, results in a finite value, positive infinity, or negative infinity. Notice that in this case, we actually do not have a direct definition of the utility function of an infinite trajectory. Instead, we define the expected utility directly. In this way, we can define the optimal expected utility of a state s over an infinite horizon to be

$$v^*_U(s) = \sup_{\pi\in\Pi} v^\pi_U(s).$$

The supremum over all policies is well-defined if for all policies, the expected utility exists. As discussed in Section 2.1, it is also desirable for the optimal values to be finite, otherwise the agent is unable to compare policies with infinite expected utilities.

It is important that the optimal expected utilities exist, since we need to determine a policy that achieves the optimal expected utilities. Figure 2.2 illustrates a case where the optimal expected utilities do not exist. An agent that starts in state s^1 receives the

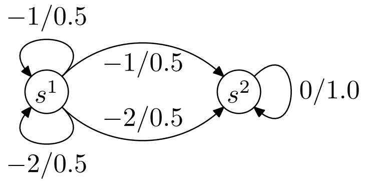


Figure 2.3: An example for the finiteness of values

following sequence of rewards for its only policy: $+1, -1, +1, -1, \ldots$, and consequently the following sequence of total rewards: $+1, 0, +1, 0, \ldots$, which oscillates. Thus, the limit in Eq. (2.6) does not exist for this policy with any utility function where $U(1) \neq U(0)$, and the optimal expected utility of state s^1 does not exist either. A similar argument holds for state s^2 as well.

It is also important that the optimal expected utilities are finite. The MDP in Figure 2.3 illustrates a case that the optimal expected utilities are infinite and the problem that it poses for DT planners. The MDP has two SD policies. Policy π_1 assigns the top action to state s^1, and policy π_2 assigns the bottom action to state s^1. For either policy, the total reward follows a geometric distribution. Suppose the utility function is $U(w) = -\gamma^w$ where $0 < \gamma < 1$. For state s^2, we have $v_U^{\pi_1}(s^2) = v_U^{\pi_2}(s^2) = -1$. For state s^1, the expected utility for policy π_1 is

$$v_U^{\pi_1}(s^1) = \sum_{t=1}^{\infty}\left[-\gamma^{(-1)t} \cdot 0.5^t\right] = -\sum_{t=1}^{\infty}\left(\frac{1}{2\gamma}\right)^t = \begin{cases} -\frac{1}{2\gamma-1}, & \frac{1}{2} < \gamma < 1 \\ -\infty, & 0 < \gamma \leq \frac{1}{2}, \end{cases}$$

and the expected utility for policy π_2 is

$$v_U^{\pi_2}(s^1) = \sum_{t=1}^{\infty}\left[-\gamma^{(-2)t} \cdot 0.5^t\right] = -\sum_{t=1}^{\infty}\left(\frac{1}{2\gamma^2}\right)^t = \begin{cases} -\frac{1}{2\gamma^2-1}, & \sqrt{\frac{1}{2}} < \gamma < 1 \\ -\infty, & 0 < \gamma \leq \sqrt{\frac{1}{2}}. \end{cases}$$

Therefore, if $0 < \gamma \leq \frac{1}{2}$, we have $v_U^{\pi_1}(s^1) = v_U^{\pi^2}(s^1) = -\infty$. Then utility theory implies that these two policies are indifferent. However, for all finite horizons T, we have $v_{U,T}^{\pi_1}(s^1) > v_{U,T}^{\pi_2}(s^2)$. Thus intuitively, policy π_1 should be preferred over policy π_2, which contradicts our earlier conclusion.

In fact, infinite expected utilities can cause problems only if the optimal expected utilities are infinite. If the optimal utilities are finite, it is not a problem if the expected utilities for a certain policy are infinite (in fact, negative infinity). This is the case when $\frac{1}{2} < \gamma \leq \sqrt{\frac{1}{2}}$, where $v_U^{\pi_1}(s^1) = -\frac{1}{2\gamma^2-1} > v_U^{\pi_2}(s^1) = -\infty$, and thus policy π_1 is preferred over policy π_2. In fact, policy π_1 is an optimal policy for this problem.

This example also shows that this problem can exist for some utility functions, but not some other utility functions. For example, if $\sqrt{\frac{1}{2}} < \gamma < 1$, then the expected utilities for both policies are finite, and policy π_1 is preferred over π_2. The case is similar for the risk-neutral utility function $U(w) = w$.

2.2.2.3 Discounting

In this thesis, we use the total undiscounted reward. In the literature of risk-neutral planning objectives, it is popular to use the total discounted reward if the planning horizon is infinite. The total discounted reward is a summarization function defined for infinite horizon problems as

$$U_{\text{sum},\beta}(r_0, r_1, \ldots, r_t, \ldots) = \sum_{t=0}^{\infty} \beta^t r_t,$$

where $\beta \in (0, 1)$ is a discount factor. Under Condition 2.1 (Finite Model), this infinite summation is well-defined and also bounded. The expected (risk-neutral utility of the) total discounted reward of the state s for the policy π is defined as

$$v_\beta^\pi(s) = E^{s,\pi}\left[\sum_{t=0}^{\infty} \beta^t r_t\right],$$

and the optimal expected (risk-neutral utility of the) total discounted reward of state s is

$$v_\beta^*(s) = \sup_{\pi \in \Pi} v_\beta^\pi(s).$$

It is straightforward to generalize discounting to risk-sensitive planing objectives. The expected utility of total discounted reward of the state s for the policy π under a risk-sensitive planning objective with the utility function U is defined as

$$v_{U,\beta}^\pi(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{\infty} \beta^t r_t\right)\right],$$

and the optimal expected utility of the state s is

$$v_{U,\beta}^*(s) = \sup_{\pi \in \Pi} v_{U,\beta}^\pi(s).$$

Discounting is a common technique used in the literature so that under Condition 2.1 and for all possible trajectories, the total discounted rewards are finite, therefore the values always exist and are finite. Using discounting also makes it simpler to construct efficient planners for MDPs and characterize how good the resulting plans are. Without discounting, these problems are more subtle and more difficult.

We discuss total discounted reward for two purposes. First, our approach will make use of some planners that are developed for discounted risk-neutral planning objectives. Second, we will review some results in the literature that deal with the planning objective of maximizing the expected risk-sensitive utility of the total discounted reward. However, this thesis does not include new results under risk-sensitive planning objectives with discounting, as explained next.

We prefer not to use discounting in our discussion of risk-sensitive planning objectives for two reasons. First, discounting expresses a temporal preference that is not fully justified in AI planning. Discounting means that receiving positive rewards now is better than receiving them later, and receiving negative rewards now is worse than receiving them later (Keeney and Raiffa, 1976). In MDPs, discounting is often implemented geometrically as a discount factor $\beta \in (0, 1)$, and the reward received t time steps later is discounted by the factor β^t. Although discounting is a reasonable preference structure, its applicability and the particular form of geometric discounting have been questioned in the literature (Harvey, 1995; Laibson, 1994; Whittle, 1996). AI planning tasks often have a relatively small time span (hours) where temporal preference is not prominent. In many (risk-neutral) applications people use discount factors very close to one only for the mathematical convenience of discounting. This is the fundamental reason why we are not in favor of discounting. The second reason is more pragmatic. If discounting is used under risk-neutral planning objectives, there exist SD policies that are optimal, the same as in the undiscounted case. Thus discounting does not require additional structural complexity to represent the optimal policies that we seek. For risk-sensitive planning objectives, this is not the case. For exponential utility functions, it is known that the optimal policy is time-dependent if discounting is used (Jaquette, 1976; Chung and Sobel, 1987), while there exists an SD policy that is optimal if discounting is not used (Ávila-Godoy, 1999; Cavazos-Cadena and Montes-de-Oca, 2000a). Similarly, for more general risk-sensitive utility functions, existing results show that optimal policies for discounted problems have more complex structures (White, 1987) than those for undiscounted problems (see Chapter 4 and Chapter 5). Therefore, we do not use discounting in our planning objectives.

2.2.2.4 Values, Optimal Policies, and Notation

To simplify our terminology, we refer to the expected utility of the state s for the policy π as the value of the state s for the policy π, and also refer to the optimal expected utility of the state s as the optimal value of the state s.

According to utility theory (see Section 2.1.1), we seek optimal or ϵ-optimal policies. An optimal policy for state s is a policy whose value of s equals the optimal value of s. An optimal policy is a policy that is optimal for all states. It is possible that the optimal values cannot be achieved by any policy. In this case, we are interested in ϵ-optimal policies. An ϵ-optimal policy for state s is a policy whose value is lower than the optimal value of s by at most ϵ, and an ϵ-optimal policy is a policy that is ϵ-optimal for all states. We are also interested in ϵ-optimal policies if the optimal ones are difficult to find.

For the convenience of later discussion, we list in Table 2.1 the notation for different planning objectives, as well as the notation for values and policies under these planning objectives. For optimal policies and ϵ-optimal policies, we do not use a different notation to distinguish whether it is with respect to particular initial states, rather we rely on the context to make this distinction. We use the following convention when choosing the notation: T indicates the planning horizon, and it is omitted for infinite horizon problems; β indicates the discount factor, and it is omitted for undiscounted problems; U indicates the risk-sensitive utility function, and it is omitted for linear (risk-neutral) utility functions. For the notation for planning objectives, MER indicates the risk-neutral planning objectives, and MEU indicates risk-sensitive planning objectives. We include the planning objectives with exponential utility functions, which are distinguished by the subscript $_{\exp}$, since they will be discussed in detail in Chapter 3. We also include the infinite horizon discounted

Table 2.1: Notation for planning objectives, values, and policies

Planning Objective				Values for Policy π	Optimal Values	Optimal Policy	ϵ-Optimal Policy
Utility Function	Horizon	Discounting	Notation				
Risk-Neutral (Linear)	Finite	No	MER_T	v_T^π	v_T^*	π_T^*	$\pi_T^{*,\epsilon}$
	Infinite	No	MER	v^π	v^*	π^*	$\pi^{*,\epsilon}$
		Yes	MER_β	v_β^π	v_β^*	π_β^*	$\pi_\beta^{*,\epsilon}$
Exponential	Finite	No	$\mathsf{MEU}_{\mathrm{exp},T}$	$v_{\mathrm{exp},T}^\pi$	$v_{\mathrm{exp},T}^*$	$\pi_{\mathrm{exp},T}^*$	$\pi_{\mathrm{exp},T}^{*,\epsilon}$
	Infinite	No	$\mathsf{MEU}_{\mathrm{exp}}$	v_{exp}^π	v_{exp}^*	π_{exp}^*	$\pi_{\mathrm{exp}}^{*,\epsilon}$
		Yes	$\mathsf{MEU}_{\mathrm{exp},\beta}$	$v_{\mathrm{exp},\beta}^\pi$	$v_{\mathrm{exp},\beta}^*$	$\pi_{\mathrm{exp},\beta}^*$	$\pi_{\mathrm{exp},\beta}^{*,\epsilon}$
General Risk-Sensitive (Nonlinear)	Finite	No	MEU_T	$v_{U,T}^\pi$	$v_{U,T}^*$	$\pi_{U,T}^*$	$\pi_{U,T}^{*,\epsilon}$
	Infinite	No	MEU	v_U^π	v_U^*	π_U^*	$\pi_U^{*,\epsilon}$
		Yes	MEU_β	$v_{U,\beta}^\pi$	$v_{U,\beta}^*$	$\pi_{U,\beta}^*$	$\pi_{U,\beta}^{*,\epsilon}$

planning objectives, where planners under the MER_β objective will be used as the basis for our transformation-of-algorithms approach, and we will present in Section 2.4 results from the literature under MEU_β objectives.

2.2.3 Basic Problems Concerning Planning with MDPs

Operations research studies the following basic problems of MDP models. Solutions to these problems provide the basis for solving AI planning problems represented as MDPs. In this thesis, we also need to explore these problems for risk-sensitive planning objectives, especially for MDP models with general rewards.

2.2.3.1 Existence and Finiteness of Optimal Values

The first problem is the existence and finiteness of the optimal values under a given planning objective, which in turn depend on the existence and finiteness of values for a given policy. We have shown that for a finite model, these properties are guaranteed for finite horizon problems, and for infinite horizon problems with discounting. But for problems under the objective of maximizing the expected risk-sensitive utility of total (undiscounted) rewards over an infinite horizon, we still need to identify what additional conditions are required to ensure the existence and finiteness properties, as we have illustrated using examples from Figure 2.2 and Figure 2.3. If the values for all policies exist, then the optimal values exist. If, in addition, the values of all states for all policies are bounded from above, the optimal values are finite.

2.2.3.2 Optimality Equations

The second problem is the optimality equations. Optimality equations represent the relationship among the optimal values of different states. This relationship is important because the agent transitions among states, and the values of different states are interdependent. Therefore, we need to identify the optimality equations under risk-sensitive planning objectives. Besides, a solution to the optimality equations is not guaranteed to yield the optimal values. So we also need to identify conditions that ensure that the solution gives the optimal values.

2.2.3.3 Optimal Policies

The desired result from solving an MDP model is an optimal or ϵ-optimal policy, not just optimal values. Therefore, the next problem is about optimal policies, especially the existence of an optimal policy, the relationship of an optimal policy to the optimal values, and the existence of an optimal policy of a special structure. The existence of an optimal policy means whether the optimal values can be achieved by a policy, and whether they can be achieved simultaneously for all states. If an optimal policy exists, we also need to know the relationship between an optimal policy and the optimal values, so that we can judge based on the optimal values whether a policy is optimal. In addition, it is desired to have an optimal policy in a restricted class of policies, such as Π^{MD} or Π^{SD}, so that the structure of an optimal policy is simplified. If a K policy is optimal, we say this policy is K-optimal, where $K \in \{\mathrm{HR}, \mathrm{HD}, \mathrm{MR}, \mathrm{MD}, \mathrm{SR}, \mathrm{SD}\}$.[4]

[4] Do not confuse a K-optimal policy with a policy that is optimal within the class of K policies, to which we will not give a special name.

If there exists no optimal policy, we will instead seek an ϵ-optimal policy to approximate the optimal values to within a given error $\epsilon > 0$. In this case, we need to answer a set of questions similar to those for optimal policies. If a K policy is ϵ-optimal, we say this policy is K-ϵ-optimal, where $K \in \{\mathrm{HR}, \mathrm{HD}, \mathrm{MR}, \mathrm{MD}, \mathrm{SR}, \mathrm{SD}\}$. Notice that if there exists an K-optimal policy, there also exists a K-ϵ-optimal policy, simply because an optimal policy is also ϵ-optimal for any choice of $\epsilon > 0$.

2.2.3.4 Computational Procedures

The last and the most important problem for AI planning concerns computational procedures for finding an optimal or ϵ-optimal policy. The most important solution procedures for MDPs are dynamic programming methods.[5] Basic dynamic programming methods for solving MDPs include backward induction for finite horizon problems, and value iteration and policy iteration for infinite horizon problems. We thus consider generalizations of these methods for risk-sensitive planning objectives.

Value functions play an important role when solving MDPs. A value function is a mapping from states to extended real numbers $v : S \mapsto \bar{\mathbb{R}}$. The optimal values form an optimal value function. Value functions are important since if the optimal value function is known, an optimal policy can often be retrieved in a greedy fashion.

2.3 Risk-Neutral Planning Objectives and Results

In this section, we review basic results under risk-neutral planning objectives. These results are important since they are the basis for our discussion of risk-sensitive planning objectives. First, for exponential utility functions, an important class of risk-sensitive utility functions, risk-neutral planners serve as the starting point of our transformation-of-algorithms approach. Second, for general risk-sensitive utility functions, results for countable state space models under risk-neutral planning objectives will be used directly since a state-augmentation approach is taken. This section reviews basic results for solving MDPs under risk-neutral planning objectives, which are mainly taken from (Puterman, 1994).

[5] Another solution method is linear programming. Since most of the DT planners use dynamic programming methods, we will not discuss linear programming in this thesis.

A risk-neutral utility function is a linear function, which is equivalent to the identity function since utility functions are unique up to a positively linear transformation. With an identity utility function, we have for any random variables $\mathbf{x}$ and $\mathbf{y}$,

$$E[\mathbf{x}+\mathbf{y}] = E[\mathbf{x}] + E[\mathbf{y}] \tag{2.7}$$

which holds whether or not $\mathbf{x}$ and $\mathbf{y}$ are independent. This decomposition is the basis of the results and computational methods under risk-neutral planning objectives.

The above decomposition also explains why the MDP literature usually uses the reward model $r(s,a)$ for risk-neutral planning objectives. In fact, the most general reward model is that the immediate reward received for a transition (s,a,s') forms a distribution on $\mathbb{R}$, denoted as $F_r(r|s,a,s')$. But we can show that for risk-neutral planning objectives, it is sufficient to use the reward model $r(s,a)$. Consider state s and action $a \in A_s$. For any random total future reward $\mathbf{r}$, which can be dependent upon the next-time state s', we have

$$E^{s,a}[r_0 + \mathbf{r}] = E^{s,a}[r_0] + E^{s,a}[\mathbf{r}]$$

according to Eq. (2.7). Since the first term of the above expression does not involve the next-time state s' and only depends on s and a, the reward model $r(s,a) = E^{s,a}[r_0]$ is sufficient to capture all kinds of rewards that occur after performing a in s. Moreover, we can define

$$\begin{aligned} r(s,a) = E^{s,a}[r_0] = E^{s,a}_{s'}\Big[E^{s,a}\left[r_0 \,\middle|\, s'\right]\Big] &= \sum_{s' \in S} E^{s,a}\left[r_0 \,\middle|\, s'\right] \cdot P(s'|s,a) \\ = \sum_{s' \in S} \int r F_r(dr|s,a,s') \cdot P(s'|s,a). \end{aligned}$$

But for consistency and later convenience, we still use the reward model $r(s,a,s')$ in this thesis. We will see in Chapter 3 and Chapter 4 that the reward model $r(s,a)$ is not sufficient for risk-sensitive planning objectives.

2.3.1 Finite Horizon

For finite horizon problems under Condition 2.2 (Countable States), the values for all policies exist and are bounded, since the number of possible trajectories starting from one state is

Algorithm 2.1 Backward Induction for Finite Models under the MER_T Objective

```
$v^*_T, \pi^*_T$ = BackwardInduction($M, T$)
Input:
    • $M = (S, A, P, r)$, a finite MDP model;        • $T$, a planning horizon;
Output:
    • $v^{*,t}_T$, the optimal values;               • $\pi^*_T = (d^*_0, d^*_1, \cdots, d^*_{T-1})$, an MD-optimal policy;

t ← T;
for all $s \in S$ do
    $v^{*,t}_T(s) \leftarrow 0$;
end for
for $t \leftarrow T - 1$ downto 0 do
    for all $s \in S$ do
        $v^{*,t}_T(s) \leftarrow \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v^{*,t+1}_T(s')\right]$;
        select $d^*_t(s) \in \arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v^{*,t+1}_T(s')\right]$;
    end for
end for
```

finite. Therefore the optimal values always exist and are finite. The system of optimality equations is given by

$$\begin{aligned} v^T_T(s) &= 0, && s \in S, \\ v^t_T(s) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v^{t+1}_T(s')\right], && s \in S,\ t \in \mathbb{N} \text{ where } 0 \leq t < T. \end{aligned}$$

Since the optimality equations are actually recursive definitions, the optimality equations have a unique solution such that for all states $s \in S$, $v^0_T(s) = v^*_T(s)$ (Puterman, 1994, Theorem 4.5.1). In fact, for all states $s \in S$ and all $t \in \mathbb{N}$ where $0 \leq t \leq T$, it holds that $v^t_T(s) = v^*_{T-t}(s)$, since the transition probabilities and the reward function do not change over time. Moreover, under Condition 2.2, there exists an MD-optimal policy (Puterman, 1994, Proposition 4.4.3a).

If Condition 2.1 (Finite Model) holds, then the optimality equations can be solved using the backward induction (BI) procedure Algorithm 2.1 (BackwardInduction), which simply follows from the optimality equations. The BI procedure also finds an MD-optimal policy. In principle, the procedure is also correct if Condition 2.2 holds, but it is computationally infeasible since the number of states is infinite. However, the version with an infinite state space is the basis for our discussion of risk-sensitive planning objectives with general utility functions in Chapter 4.

2.3.2 Infinite Horizon with Discounting

If the planning horizon is infinite, there are subtle differences between the MER_β and MER objectives, so we discuss them separately.

Most of the existing DT planners use discounting, and these planners are the starting points of our transformation-of-algorithms approach for risk-sensitive planning with exponential utility functions. Therefore, we review results for the MER_β objective in this section.

For discounted infinite horizon problems under Condition 2.1 (Finite Model), the values for all policies exist and are bounded, therefore the optimal values exist and are finite. The optimal values satisfy the following system of optimality equations:

$$v_\beta(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a)\big[r(s,a,s') + \beta v_\beta(s')\big], \qquad s \in S, \tag{2.8}$$

where $\beta \in (0,1)$, and the optimal values v^*_β form the unique solution. Under Condition 2.1, there exists an SD-optimal policy (Puterman, 1994, Theorem 6.2.10a).

An SD-optimal policy can be retrieved if the optimal values are known according to the following result.

Theorem 2.1 (Theorem 6.2.7b in Puterman, 1994)**.** *Under Condition 2.1, if an SD policy π satisfies*

$$v^*_\beta(s) = \sum_{s' \in S} P(s'|s,\pi(s)) \left[r(s,\pi(s),s') + \beta v^*_\beta(s')\right], \qquad s \in S, \tag{2.9}$$

then π is optimal under the MER_β objective where $\beta \in (0,1)$.

A stationary policy satisfying Eq. (2.9) is called conserving. Obviously, an optimal policy must be conserving. Furthermore, if conserving policies are also optimal, and the optimal values are known, then a greedy policy with respect to the optimal values is optimal. A greedy policy π^v_β with respect to a value function v is an SD policy such that

$$\pi^v_\beta(s) \in \arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + \beta v(s')\right], \qquad s \in S.$$

The concepts of greedy policies and conserving policies can be generalized to other planning objectives if we change the equations involved accordingly in a similar way in which the optimality equations change.

Algorithm 2.2 Value Iteration Procedure for Finite Models under the MER_β Objective

$v, \pi = \mathsf{ValueIteration}(M, \beta, \epsilon)$

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- β, a discount factor, $0 < \beta < 1$;
- ϵ, an accuracy parameter, $\epsilon > 0$;

Output:

- $v_\beta^{*,\epsilon}$, an ϵ-optimal value function;
- $\pi_\beta^{*,\epsilon}$, an SD-ϵ-optimal policy;

```
 1: initialize $v_0$ to be an arbitrary finite vector;
 2: $t \leftarrow 0$;
 3: repeat
 4:     for all $s \in S$ do
 5:         $v^{t+1}(s) \leftarrow \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + \beta v^t(s')\right]$;
 6:     end for
 7:     $t \leftarrow t + 1$;
 8: until $\|v^t - v^{t-1}\| \leq \frac{1-\beta}{2\beta}\epsilon$;
 9: $v_\beta^{*,\epsilon} \leftarrow v^t$;
10: for all $s \in S$ do
11:     select $\pi_\beta^{*,\epsilon}(s) \in \arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + \beta v_\beta^{*,\epsilon}(s')\right]$;
12: end for
```

The value iteration (VI) procedure Algorithm 2.2 ($\mathsf{ValueIteration}$) can be viewed as a generalization of backward induction to infinite horizon problems. Value iteration uses a value function to approximate the optimal values. Value iteration starts with an initial value function, and gradually improves the value function as shown on Line 5 until it is sufficiently close to the optimal values. When the approximation is close enough, an ϵ-optimal policy can be obtained by acting greedily as shown on Line 11.

The policy iteration (PI) procedure Algorithm 2.3 ($\mathsf{PolicyIteration}$), on the other hand, gradually improves the current policy until it is optimal. Policy iteration starts with an arbitrary policy, and alternates between policy evaluation and policy improvement until the policy is optimal. The policy evaluation step Line 4 obtains the values for the current policy, and the policy improvement step Line 6 tries to obtain a better policy based on the values resulting from the policy evaluation step.

It is worth noting that it is not necessary to have a non-empty set of goal states under the MER_β objective. On the other hand, if there exist goal states, the computational procedures will calculate the correct values for goal states (namely zero) starting from an arbitrary value function (for VI) or an arbitrary policy (for PI). Therefore, the optimality equations and

Algorithm 2.3 Policy Iteration Procedure for Finite Models under the MER_β Objective

v, π = PolicyIteration(M, β)

Input:
- $M = (S, A, P, r)$, a finite MDP model;
- β, a discount factor vector, $0 < \beta < 1$;

Output:
- v^*_β, the optimal values;
- π^*_β, an SD-optimal policy;

```
initialize $\pi^0$ to be an arbitrary SD policy;
$t \leftarrow 0$;
repeat
    solve for $v^t(s)$ from the system of equations
    $v_\beta(s) = \sum_{s' \in S} P(s'|s, \pi^t(s)) \left[ r(s, \pi^t(s), s') + \beta v_\beta(s') \right], \qquad s \in S$;
    for all $s \in S$ do
        select $\pi^{t+1}(s) \in \arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \left[ r(s, a, s') + \beta v^t(s') \right]$,
        setting $\pi^{t+1}(s) = \pi^t(s)$ if possible;
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi^*_\beta \leftarrow \pi^t$;
$v^*_\beta \leftarrow v^{t-1}$;
```

computational procedures under the MER_β objective do not need to mention goal states explicitly. But for the undiscounted objective MER, special treatments are needed for goal states, as we will see next.

2.3.3 Infinite Horizon without Discounting

The undiscounted risk-neutral planning objective MER is a lot subtler than the MER_β objective. However, results for the MER objective provide a starting point for our transformation-of-algorithm approach for risk-sensitive planning with exponential utility functions in Chapter 3, and also the foundation for our discussion of risk-sensitive planning with general risk-sensitive utility functions in Chapter 4. Therefore, this objective deserves a detailed discussion.

For undiscounted infinite horizon problems, the value of a state $s \in S$ under a given policy $\pi \in \Pi$ is defined as

$$v^\pi(s) = \lim_{T \to \infty} E^{s,\pi} \left[\sum_{t=0}^{T-1} r_t \right] = \lim_{T \to \infty} v^\pi_T(s). \tag{2.10}$$

Such a limit may not exist or can be infinite.

There are two classes of undiscounted infinite horizon problems, one with a non-empty set of goal states, and one without. We are interested in problems with goal states since AI planning problems often have a non-empty set of goal states. But results for problems with goal states rely on the assumption that a goal state can be reached, which may not be known in advance for AI planning problems. Therefore, we are also interested in problems without goal states since the results do not rely on the above assumption. We are interested in problems without goal states also because there exist results for countable state space models, which will be used in our discussion of risk-sensitive planning with general utility functions. We first consider problems with goal states in Section 2.3.3.1, then consider problems without explicit goal states in Section 2.3.3.2.

2.3.3.1 Problems with Explicit Goal States

In AI planning, we are often interested in problems with goal states. These problems are also viewed as stochastic generalizations of shortest path problems (Bertsekas and Tsitsiklis, 1991). We assume that Condition 2.1 (Finite Model) holds when considering problems with goal states.

For problems with goal states, we are interested in policies that can reach a goal state. Such a stationary policy is called proper. Formally, a stationary policy π is proper if and only if under this policy, there exists a finite number n such that the value

$$\rho_n^\pi = \max_{s \notin G} P^{s,\pi}(s_n \notin G) < 1, \tag{2.11}$$

where s_n is the state at decision epoch n. A stationary policy that is not proper is an improper policy. If a goal state cannot be reached from s under an improper policy π, we say π is improper at s.

Intuitively, a proper policy is a policy under which a goal state can be reached with a positive probability from any state in finite time. Consequently, it can be proved that a goal state can be reached with probability one from any state under a proper policy. It is natural and necessary to have the following condition in the following discussions.

Condition 2.3 (Proper Policy)**.** There exists at least one proper policy.

Condition 2.3 concerns the reachability of a goal state, regardless of the reward function of the model. It is possible that there exist states from which a goal state cannot be reached, but an optimal policy may never enter those states starting from a given set of initial states. Then this condition is overly restrictive. However, in this case, we can restore Condition 2.3 by deleting such states from the model. One way to guarantee this condition is to delete from the model those states and actions that lead to unavoidable infinite cycles, which can be done through the Selective State-Deletion procedure given in (Koenig and Liu, 2002) if the state space can be enumerated.

For a problem with goal states, the value of a goal state is always zero for any policy since only zero rewards can occur. If we assume that the optimal values exist, we have the following system of optimality equations

$$\begin{aligned} v(s) &= 0, && s \in G \\ v(s) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v(s') \right], && s \notin G. \end{aligned}$$

However, without additional conditions, the optimality equations may not have a solution, or may have multiple solutions. The following conditions eliminate these possibilities.

2.3.3.1.1 Bertsekas's Condition for Improper Policies

Bertsekas (2001, Chapter 2) proposed the following condition that prevents an improper policy from being optimal.

Condition 2.4 (Infinite Values for Improper Policies)**.** For every improper policy π, the value $\lim_{T \to \infty} v_T^\pi(s) = -\infty$ for at least one state s.

One important special case for AI planning is $r(s,a,s') < 0$ for all transitions (s,a,s') where $s \notin G$, which is the action penalty representation (Dean *et al.*, 1995), that is, penalizing all action executions until a goal state is reached. In addition, we can also allow for models that give positive rewards only for transitions resulting in a goal state. However, Condition 2.4 can be hard to check in general.

Bertsekas (2001, Chapter 2) obtained the following results under Condition 2.1, Condition 2.3, and Condition 2.4: the optimal values exist and are finite; the optimal equations have a unique solution, which is the optimal values; there exists an SD-optimal policy that is also proper; and a conserving policy is optimal.

The main computational procedures for the MER_β objective can be adapted to solve problems with goals under the MER objective (Bertsekas, 2001, Chapter 2). In both value iteration and policy iteration, we need to fix the values of goal states to be zero. Then the value iteration procedure converges to the optimal values with any initial value vector, and the policy iteration procedure converges to a deterministic optimal proper policy if the initial policy is proper, which is often easy to obtain.

2.3.3.1.2 Bertsekas's Nonpositive Reward Condition

Under Condition 2.1 and Condition 2.3, Bertsekas (2001, Exercise 2.12) showed that results similar to those from the previous section hold if $r(s, a, s') \leq 0$ for all possible transitions where $s \notin G$. More concretely, the optimal values exist and are finite, and they are the unique solution to the optimality equations if we only consider nonpositive values. There also exists an SD-optimal proper policy, and a conserving policy is optimal. However, there exist improper policies whose values equal the optimal values. Bertsekas (2001, Exercise 2.12) mistakenly claimed that a conserving policy is also proper. A counterexample is a model with all zero rewards. Then the optimal values are all zero, and any policy is conserving and optimal, but not necessarily proper. So even if the optimal values are known, acting greedily may not result in an optimal proper policy. The computational methods are still valid, but special care is needed so that we obtain a proper policy.

2.3.3.2 Problems without (Explicit) Goal States

Undiscounted infinite horizon problems without goal states are also important since the technical conditions to ensure the existence of a proper optimal policy is not easy to check and sometimes overly restrictive. It is also important to have some knowledge about models without goal states as the basis for our later discussion of risk-sensitive planning with general risk-sensitive utility functions.

Actually, models with explicit goal states are special cases of models without goal states. For models without explicit goal states, it is still the case that the agent will eventually enter a set of recurrent states, from which no other states are reachable, but this set of states is not trivially known in advance as in models with explicit goal states.

Two special cases of undiscounted infinite horizon problems without goal states are negative and positive models, where the rewards are all nonpositive and nonnegative, respectively. Models discussed in Section 2.3.3.1.2 are special cases of negative models, and we will see the differences in results shortly. We discuss these two cases in this chapter, and models with more general rewards will be discussed in Chapter 5. For later convenience, we discuss these models under Condition 2.2 (Countable States), rather than under Condition 2.1 (Finite Model) as in Section 2.3.3.1.

Under the condition that the optimal values exist, the optimal values satisfy the optimality equation

$$v(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v(s') \right], \qquad s \in S, \tag{2.12}$$

although some values can be infinite (Puterman, 1994, Chapter 7). Unfortunately, the solution to this equation is not unique, since for example, if $v(\cdot)$ is a solution value function, then $v(\cdot) + k$ for any real number $k \in \mathbb{R}$ is also a solution value function.

2.3.3.2.1 Negative Models

In negative models, the immediate rewards are all nonpositive as shown in the following condition. AI planning problems are often modeled as negative models where the negative rewards represent action costs.

Condition 2.5 (Negative Model)**.** For all $s, s' \in S$ and all $a \in A_s$ where $P(s'|s,a) > 0$, it holds that $r(s,a,s') \leq 0$.

Since only negative or zero rewards are possible, the finite horizon values are monotonic in the planning horizon. Therefore, the expected total rewards for all policies and the optimal values exist. However, these values can be negative infinity. The following condition is needed to ensure the optimal values are finite. Notice that this condition should always go together with Condition 2.5.

Condition 2.6 (Negative Model with Finite Expected Rewards)**.** There exists a policy $\pi \in \Pi$ such that for all states $s \in S$, the value $v^\pi(s)$ is finite.

Suppose Condition 2.2 (Countable States), Condition 2.5 and Condition 2.6 hold. The optimal value v^* is the maximal nonpositive solution to the optimal equation. Under the same conditions, there exists an SD-optimal policy (Puterman, 1994, Theorem 7.3.6a). Moreover, a conserving policy is optimal (Puterman, 1994, Theorem 7.3.5).

Under Condition 2.2, Condition 2.5, and Condition 2.6, value iteration converges statewise with any initial value function v^0 such that for all states $s \in S$, $0 \geq v^0(s) \geq v^*(s)$ (Puterman, 1994, Corollary 7.3.12). The value iteration procedure can be carried out if Condition 2.1 also holds. Unfortunately, a direct generalization of policy iteration from discounted models does not work for negative models since it is possible that a suboptimal policy cannot be improved any further (Puterman, 1994, Example 7.3.4). A proper policy iteration procedure is more complex (Puterman, 1994, Section 10.4).

2.3.3.2.2 Positive Models

In positive models, the immediate rewards are all nonnegative as shown in the following condition. Positive models have been used to model problems such as gambling and optimal stopping (Puterman, 1994, Section 7.2).

Condition 2.7 (Positive Model)**.** For all $s, s' \in S$ and all $a \in A_s$ where $P(s'|s,a) > 0$, it holds that $r(s,a,s') \geq 0$.

Since all rewards are nonnegative, the finite-horizon values are monotonic with the planning horizon. Therefore, the expected total rewards for all policies and the optimal values exist. However, these values can be positive infinity. To ensure the values for all policies to be finite, we need the following condition. Under Condition 2.1 (Finite Model), this condition also implies the optimal values are finite. Similar to the negative model case, this condition always goes with Condition 2.7.

Condition 2.8 (Positive Model with Finite Expected Rewards)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the value $v^\pi(s)$ is finite.

Suppose Condition 2.2 (Countable States), Condition 2.7, and Condition 2.8 hold. The optimal value v^* is the minimal nonnegative solution to the optimality equation (Puterman, 1994, Theorem 7.2.3a).

Moreover, there exists an SD-optimal policy if Condition 2.1 holds (Puterman, 1994, Theorem 7.1.9). But otherwise, only a weaker result can be obtained (Puterman, 1994, Theorem 7.2.7): for all $\epsilon > 0$, there exists an SD policy π such that

$$v^{\pi}(s) \geq (1 - \epsilon) v^*(s), \qquad s \in S.$$

Consequently, if for all states $s \in S$, the optimal value $v^*(s)$ is bounded, then there exists an SD-ϵ-optimal policy (Puterman, 1994, Corollary 7.2.8). On the other hand, under Condition 2.2, Condition 2.7, and Condition 2.8 only, if there exists an optimal policy, then there exists an SD-optimal policy (Puterman, 1994, Theorem 7.2.11). However, a conserving policy may not be optimal (Puterman, 1994, Example 7.2.3).

Similar to the discounted case, value iteration and policy iteration can be used to solve positive models, but minor changes are needed. Under Condition 2.2, Condition 2.7, and Condition 2.8, value iteration converges statewise with any initial value function v^0 such that for all states $s \in S$, $0 \leq v^0(s) \leq v^*(s)$ (Puterman, 1994, Corollary 7.2.13). The value iteration procedure can be carried out if Condition 2.1 (Finite Model) also holds.

Under Condition 2.1, an optimal policy can also be obtained using policy iteration with subtle changes to the procedure (Puterman, 1994, Section 7.2.5). When solving the policy evaluation equations, the solution is not unique. To obtain the correct values, we need to set the values of recurrent states to zero. Since the model is reduced to a Markov chain under fixed stationary policies, a recurrent state under a stationary policy is the same as a recurrent states in a finite Markov chain. For finite Markov chains, a state is recurrent if and only if starting from this state, the expected first time of returning to this state is finite. In other words, a state is recurrent if and only if the state is in a cycle of the graph corresponding to the Markov chain. Therefore, the recurrent states can be identified using algorithms from graph theory, for example, topological sorting (Corman *et al.*, 1990).

2.3.4 Semi-MDPs

For MDP models, the decisions are made at discrete decision epochs. These decision epochs are assumed to be equally divided in time, or we assume the passage of time is irrelevant except that decisions are made at specific points of time. If it is not the case, we need to consider continuous time models, or semi-MDPs (SMDPs) (Puterman, 1994, Chapter 11).

We consider SMDPs in this section because results for SMDPs provide another starting point for our transformation-of-algorithms approach in Chapter 3. In fact, we use the variably discounted form (see below) of SMDPs instead of the original form, although the latter is more popular.

The major difference between MDPs and SMDPs is the sojourn time, or the time of staying in a state before transitioning to the next state. MDPs do not explicitly model this time since all transition times are equal or irrelevant. In SMDPs, the sojourn times to the next state follow the probability distribution $F_t(t|s, a, s')$. This sojourn time affects the decision only when we consider the discounted total reward objective.

Discounting in SMDPs is implemented through a discount rate $0 < \beta < 1$. Given a trajectory, the total discounted reward is $\sum_{t=0}^{\infty} \beta^{\sigma_t} r_t$, where $\sigma_t = \sum_{k=0}^{t-1} \tau_k$, and τ_k is the sojourn time at decision epoch k.

Parr (1998) observed that a discounted SMDP is equivalent to a variably discounted MDP (VDMDP) where the discount factor depends on the transitions. For later convenience, we refer to a VDMDP as an MDP under the $\mathsf{MER}_{\boldsymbol{\beta}}$ objective, where the boldface $\boldsymbol{\beta}$ indicates that the discount factors depend on the transitions, and this notational convention extends to value functions and optimal policies. The optimality equations for SMDPs or VDMDPs are

$$v_{\boldsymbol{\beta}}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \left[r(s, a, s') + \beta(s, a, s') v_{\boldsymbol{\beta}}(s') \right],$$

where $\beta(s, a, s')$ is a transition-dependent discount factor, and

$$\beta(s, a, s') = \int_0^\infty \beta^t F_t(dt|s, a, s').$$

The computational procedures for (regular) discounted MDPs also apply to discounted SMDPs, namely value iteration and policy iteration methods, provided that the minimal

Table 2.2: Results for finite models with risk-neutral planning objectives

Objective	MER_T	MER_β	MER			
			$G \neq \varnothing$		$G = \varnothing$	
Existence of optimal values	Yes	Yes	$\exists$ proper π; improper $\pi \Rightarrow$ $\exists s, v^\pi(s) = -\infty$	$r(s,a,s') \leq 0$	$r(s,a,s') \geq 0$	$r(s,a,s') \leq 0$
Finiteness of optimal values	Yes	Yes	Yes	$\exists$ proper π	$\forall \pi, v^\pi < +\infty$	$\exists \pi, v^\pi > -\infty$
Solution to optimality equations?	Unique	Unique	Unique	Unique	Minimal nonnegative	Maximal nonpositive
Structure of optimal policy	MD	SD	SD, proper	SD	SD	SD
Is conserving policy optimal?	Yes	Yes	Yes	Yes	No	Yes
Computational procedures	BI	VI, PI	VI, PI	VI, PI	VI, PI	VI, PI'

sojourn time is greater than zero (thus the transition-dependent discount factors are less than one) (Puterman, 1994, Chapter 11). The requirement that the minimal sojourn time is greater than zero also eliminates the possibility to have an infinite number of transitions during a finite period. However, if the minimal sojourn time can be zero (thus the transition-dependent discount factors can be one), it is not guaranteed that the optimality equations have a unique solution in general. In this case, the SMDPs (and VDMDPs) have properties similar to undiscounted MDPs, and the computational procedures are analogous to those for undiscounted MDPs with similar restrictions.

2.3.5 Summary

We summarize the results for finite models with risk-neutral planning objectives in Table 2.2, where PI' in the last entry indicates that a more complicated variant of PI is needed. For conciseness, the table should be read that the conditions needed for the finiteness property also include those for the existence property. Limited by space, some conditions are not presented in their complete form, so we should refer back to the main text for the exact conditions.

2.4 Related Work: Risk-Sensitive Planning

In this section, we briefly review existing results from operations research and DT planning for risk-sensitive planning objectives and other planning objectives based on the MEU principle.

2.4.1 MDPs with Exponential Utility Functions

Exponential utility functions are the most widely used risk-sensitive utility function (Corner and Corner, 1995). Exponential utility functions have the form

$$U_{\exp}(w) = \begin{cases} \gamma^w, & \gamma > 1 \\ -\gamma^w, & 0 < \gamma < 1. \end{cases}$$

The risk parameter γ indicates the degree of risk seekingness or risk aversion. If $\gamma > 1$, the agent is risk-seeking, and if $0 < \gamma < 1$, the agent is risk-averse. A more detailed discussion about exponential utility functions is given in Section 3.1.

2.4.1.1 Finite Horizon Expected Exponential Utility of Total Rewards

The finite horizon problem was first formulated in (Howard and Matheson, 1972). We refer to this planning objective as $\mathsf{MEU}_{\exp,T}$. They studied how to find optimal policies in the class of MD policies for finite horizon problems, and obtained the system of optimality equations, which has a unique solution. Based on the optimality equations, they showed that the finite horizon problems can be solved using a backward induction procedure.

Although an argument similar to that for the risk-neutral case can be used to show that under Condition 2.2 (Countable States), an optimal policy in the class of MD policies is also optimal in the class of all policies, a rigorous proof had not been made explicit until quite recently (Ávila-Godoy *et al.*, 1997). Coraluppi and Marcus (1996) also claimed without proof that there is an MD policy that is optimal in the class of HD policies.

2.4.1.2 Infinite Horizon Expected Exponential Utility of Total Discounted Rewards

The planning objective of maximizing the expected exponential utility of total discounted rewards over an infinite horizon, referred to as $\mathsf{MEU}_{\exp,\beta}$, is an extension of discounting from the risk-neutral case to the case of exponential utility functions.

Jaquette (1973) is the first study of the $\mathsf{MEU}_{\exp,\beta}$ objective. He studied the risk-averse case as a tool for studying the so-called moment optimal objective. He showed that for finite models, there exists a threshold value $\gamma_0 \in (0, 1)$ such that for all $\gamma \in [\gamma_0, 1)$, there is a stationary policy that is optimal in the class of MD policies under the $\mathsf{MEU}_{\exp,\beta}$ objective. Such a policy is also moment optimal at the same time, that is, it lexicographically maximizes the sequence of signed moments of the total discounted reward, where the sign is positive (negative) if the order of the moment is odd (even). Moreover, Jaquette (1973) presented a computation-intensive procedure for finding a moment optimal SD policy. In fact, the procedure finds all moment optimal policies, based on which the threshold risk parameter γ_0 can be determined. Jaquette (1976) further showed that for arbitrary risk-averse exponential utility functions, the policy that is optimal in the class of MD policies is non-stationary in general. He showed that such a policy is ultimately stationary, that is, the policy has a stationary tail, and that the stationary tail must be a moment optimal policy. He also presented a method for constructing an MD policy that is optimal in the class of MD policies for risk-averse agents, provided a moment optimal SD policy and the threshold risk parameter γ_0 are known. In this case, the problem is reduced to a finite horizon problem where the planning horizon is a function of γ_0, γ and β, and the desired policy can be obtained using a procedure similar to backward induction. In these studies, there is no reference to optimality equations.

Chung and Sobel (1987) studied this objective for risk-averse agents and finite models with nonnegative rewards, while the optimization is restricted to the class of MD policies. They first obtained the system of optimality equations. Different from the risk-neutral case, the system has an infinite number of equations. They showed that under their assumptions, a conserving policy (that is, a greedy policy with respect to the optimal values) is optimal in the class of MD policies, and that value iteration converges to the optimal values. They further pointed out that the result still holds for risk-seeking agents and finite models with nonpositive rewards.

Coraluppi and Marcus (1996) also presented the same system of optimality equations for risk-averse agents, by referring to Chung and Sobel (1987) for its correctness, and claimed it remains valid if the optimization is over HD policies. However, in subsequent discussions, they implicitly assume the rewards are nonpositive, therefore their argument was not well-founded until (Ávila-Godoy *et al.*, 1997) proved the validity of the system of optimality equations (for all policies). Coraluppi and Marcus (1996) discussed approximating optimal policies using finite horizon problems under the implicit assumption of nonpositive rewards. The method is a variant of the backward induction procedure.

Ávila-Godoy *et al.* (1997) showed that for countable state space models with bounded rewards (thus including finite models) and for any exponential utility function, the optimal values satisfy the same system of optimality equations. They also showed that if the model has nonpositive rewards, then there exists an MD-optimal policy, a conserving policy is optimal, and value iteration converges to the optimal values, regardless of whether the agent is risk-averse or risk-seeking.

2.4.1.3 An Alternative Discounted Objective Using Exponential Utility Functions

Another approach to extending discounting to infinite horizon problems with exponential utility functions was taken in (Porteus, 1975) based on dynamic choice theory, where the exponential utility function is viewed as an "intra"-epoch utility function. Roughly speaking, the certainty equivalent of the current state is the sum of immediate rewards and the discounted certainty equivalent of future rewards. In this case, there still exist SD-optimal policies, unlike the discounted model discussed above. Coraluppi and Marcus (1996) showed that the corresponding dynamic programming operator is a contraction mapping. Eagle (1975) took a similar approach and showed there also exists an SD-optimal policy for a slightly different objective. All these works assumed that the agent is risk-averse and the rewards are nonpositive.

2.4.1.4 Infinite Horizon Expected Exponential Utility of Total Rewards

Similar to the case of the MER objective, the expected utility of (undiscounted) total rewards using exponential utility functions may not exist, or may be infinite, since the value is defined as the limit of finite horizon values as the horizon approaches infinity. So we need to specify conditions under which such values exist and are finite. We refer to this objective as $\mathsf{MEU}_{\text{exp}}$.

We give a brief overview of results for the $\mathsf{MEU}_{\text{exp}}$ objective. More details are given in Section 3.2 since these results are more related to our transformation-of-algorithms approach to solving large-scale problems under the $\mathsf{MEU}_{\text{exp}}$ objective.

2.4.1.4.1 Problems with Explicit Goal States

Denardo and Rothblum (1979) discussed this objective for finite models with goal states in the class of SD policies, and provided a linear programming formulation for solving such problems.

For risk-averse agents, Denardo and Rothblum (1979) showed that if there exists an SD policy π for which $v^{\pi}_{\text{exp}}(s)$ is finite for all states, and if for any improper policy π', $v^{\pi'}_{\text{exp}}(s)$ is infinite for at least one state, then the following results hold: an optimal policy (in the class of SD policies) exists and is proper, the optimal solution to the linear program equals the optimal values, a conserving policy is optimal, and the optimal values are the unique negative solution to the system of optimality equations. They also showed that the required set of sufficient conditions can be checked once the linear program is solved.

For risk-seeking agents, Denardo and Rothblum (1979) showed that if for any SD policy π, $v^{\pi}_{\text{exp}}(s)$ is finite for all states, then the following results hold, which are similar to those in the risk-averse case: an SD-optimal policy exists, the optimal solution to the linear program equals the optimal values, a conserving policy is optimal, and the optimal values are the unique solution to the above optimality equations. The required sufficient condition can be checked by solving an auxiliary linear program. However, an SD-optimal policy is proper only when $v^{*}_{\text{exp}}(s) > 0$ for all states.

Patek (2001) considered risk-averse agents and finite state space models with strictly negative rewards, under the condition that there exists a proper policy with finite values. He showed that the following results hold: there is an SD policy that is optimal in the class of MD policies, the optimality equations have a unique solution, and a conserving policy is optimal. He also showed that value iteration and policy iteration can be used to solve such problems. Patek's conditions, however, are obviously implied by the conditions of (Denardo and Rothblum, 1979).

2.4.1.4.2 Problems without Goal States

Assuming the values exist for all policies, Ávila-Godoy (1999) showed that the optimal values for finite models must satisfy the system of optimality equations. Cavazos-Cadena and Montes-de-Oca (2000a,b) considered finite models with nonnegative rewards (Positive Models) and Ávila-Godoy (1999) considered finite models with nonpositive rewards (Negative Models). See Section 3.2.1 for more details.

2.4.1.5 Average Certainty Equivalent of Expected Exponential Utility

Another line of active research concerning exponential utility functions is the planning objective of maximizing the average certainty equivalents. For a given policy π, the average certainty equivalent of state s is defined as

$$\lim_{T\to\infty}\frac{1}{T}\;U_{\exp}^{-1}\left(E^{\pi,s}\left[U_{\exp}\left(\sum_{t=0}^{T-1} r_t\right)\right]\right)=\lim_{T\to\infty}\frac{1}{T}\;\log_\gamma\left(E^{\pi,s}\left[\gamma^{\sum_{t=0}^{T-1} r_t}\right]\right).$$

Howard and Matheson (1972) is the first work on this planning objective. They developed a policy iteration procedure for finite models that are aperiodically recurrent[6]. This planning objective was later studied for more general models, including countable state space models and semi-Markov models, under various technical conditions in (Marcus *et al.*, 1997; Hernández-Hernández and Marcus, 1996, 1999; di Masi and Stettner, 1999; Cavazos-Cadena and Fernández-Gaucherand, 1999; Brau-Rojas, 1999; Chawla, 2000). However, this objective does not fit into the MEU framework for decision-theoretic planning.

[6] An MDP is aperiodically recurrent if and only if under any stationary policy, the induced Markov chain is aperiodically recurrent. A Markov chain is aperiodically recurrent if and only if all states are recurrent and there does not exist t_0 such that for any (decision) epoch t and any trajectory, $s_t = s_{t+t_0}$.

2.4.1.6 Representation Transformation

Koenig and Simmons (1994a,b) and Koenig (1997) considered exponential utility functions in the context of AI planning. They took a representation transformation approach. Ideally, in this approach, a planning problem under the $\mathsf{MEU}_{\text{exp}}$ objective can be transformed into an equivalent problem under the MER objective, and a planner for the MER objective can be directly used to solve the transformed problem, thus solving the original problem. Koenig (1997) considered two types of representation transformations: additive and multiplicative transformations, where the additive transformation can be viewed as a special case of the multiplicative transformation. The multiplicative transformation converts the transition probabilities into pseudo-probabilities that do not necessarily sum to one, and the agent is only rewarded for reaching a goal state in the transformed problem. Koenig (1997) showed that for models with strictly negative rewards and risk-seeking exponential utility functions, as well as models with strictly positive rewards and risk-averse exponential utility functions, the transformed problems have an interpretation where the pseudo-probabilities are legitimate probabilities, and the original problem is transformed into a legitimate risk-neutral planning problem. Therefore computational methods for the risk-neutral objective, such as value iteration and policy iteration, can be directly used to solve problems under the $\mathsf{MEU}_{\text{exp}}$ objective. He also showed that, for acyclic problems with arbitrary rewards, the transformation can be used in a nominal sense, and methods for risk-neutral objectives can also be used directly. However, for more general models, the representation transformation method does not guarantee finding the correct result. Koenig does not discuss the applicability of the representation transformation approach to other decision-theoretic planning methods. A detailed discussion about the applicability of this approach is given in Section 3.3.

2.4.2 MDPs with Target-Level Utility Functions

Besides exponential utility functions, target-level utility functions form the only other extensively studied class of risk-sensitive utility functions (Bouakiz, 1985; White, 1993; Bouakiz and Kebir, 1995; Yu *et al.*, 1998; Wu and Lin, 1999; Ohtsubo and Toyonaga, 2002). In some applications, the agent's objective is to maximize the probability that the total discounted or undiscounted reward exceeds a target level a. This planning objective can be recast as an objective of maximizing the expected utility with a target level utility function, which is a nonlinear utility function defined as

$$U_{\text{tgt}}(w) = \begin{cases} 0, & w < a \\ \\ 1, & w \geq a \end{cases}.$$

In other words, the MEU objective with a target-level utility function models a stochastic extension of the "satisficing" planning objective (Simon, 1947).

Bouakiz (1985) considered the planning objective using target-level utility functions and total discounted rewards over an infinite horizon, as well as total undiscounted rewards over a finite horizon. He restricted attention to finite models with nonnegative rewards and considered HD policies only. For finite horizon problems, he showed that there exists an MD policy that is optimal in the class of HD policies, obtained the optimality equations, and provided a backward induction procedure. For infinite horizon problems, he obtained the optimality equations and provided a value iteration procedure. He also showed that there may not exist a stationary optimal policy for infinite horizon problems, but showed that under some technical conditions, there can be ultimately stationary optimal policies. Bouakiz and Kebir (1995) continued the previous work under the same assumptions. They showed that for finite horizon problems, the optimal values are the unique solution to the optimal equations, and that for infinite horizon problems, the optimal values are the minimal nonnegative solution to the optimal equations. They also initiated the approach of studying target-level objectives using an augmented models with an augmented state space, which is the cross product of the original state and the (equivalent) target level-to-go.

White (1993) extended the augmented space approach to consider finite models with more general reward models and optimization over all augmented policies, which are policies for the augmented models. His results include that the optimal values are the unique solution to the optimality equations, and that there exists an SD-optimal policy in the augmented sense. He also presented a policy iteration procedure. However, an error in (Bouakiz and Kebir, 1995) and (White, 1993), which is due to the neglected fact that an augmented policy may not be a policy for the original model, was pointed out by (Wu and Lin, 1999). This error invalidates the proof of a result concerning value iteration in (Bouakiz and Kebir, 1995), and also affects results concerning infinite horizon problems in (White, 1993). Wu and Lin (1999) further considered the target level objective for countable state space models with bounded rewards. For finite horizon problems, they showed that there exists an MD-optimal policy in the augmented sense, and that a conserving policy is optimal. For infinite horizon problems, they obtained the optimality equations for models with arbitrary rewards, and showed that there exists an SD conserving policy in the augmented sense and a conserving policy may not be optimal. They also provided a corrected proof for the result from (White, 1993) that there exists an SD-optimal policy in the augmented sense. Along this line, Ohtsubo and Toyonaga (2002) showed the optimal values are the minimal solution (among "reasonable" value functions) to the optimality equations. They further provided technical conditions under which the results of (White, 1993) are recovered. They also showed under the same set of conditions, a conserving policy is optimal.

Yu *et al.* (1998) extended these works to undiscounted problems with goal states and strictly positive rewards. They obtained the optimality equations, showed that there exists a unique solution, and showed that the problem can always be reduced to a finite horizon problem. For finite models, they showed that there exists an MD-optimal policy, and provided an algorithm using backward induction to obtain the optimal values and all MD-optimal policies. They also discussed an extension to the planning objective of simultaneously maximizing the values (that is, probabilities of reaching the target level) for all target levels in an interval of real numbers. For this extension, they showed that if an optimal policy exists, it must be an SD policy.

2.4.3 MDPs with General Risk-Sensitive Utility Functions

Besides the target-level objective, Bouakiz (1985) also studied quadratic utility functions and showed that in general, there does not exist an SD policy that maximizes the expected utility of total discounted rewards over an infinite horizon. He further studied general continuous risk-sensitive utility functions using the second order Taylor expansion of a utility function and results for quadratic utility functions, and concluded that there does not exist an SD policy that maximizes the expected utility of total discounted rewards over an infinite horizon.

White (1987) studied finite models under the objective of maximizing the expected utility of total undiscounted and discounted rewards with a general risk-sensitive utility function. He took a state augmentation approach that augments the state space with the accumulated total (discounted or undiscounted) rewards up to the current decision epoch. He provided the optimality equation for finite horizon undiscounted problems, and used a result from (Fainberg [sic], 1982)[7] to show that there exists an MD-optimal policy in the augmented sense for any given initial state. He also considered infinite horizon discounted problems, used the same result from (Fainberg [sic], 1982) to point out that there exists an augmented MD policy that is ϵ-optimal, and sketched a finite horizon approximation procedure for a set of continuous (Lipschitz continuous) risk-sensitive utility functions. (These claims are dubious, though. His original statement was that MD policies and augmented MD policies are "sufficient" for finite horizon and infinite horizon discounted problems, respectively. The context suggested he meant optimality. However, Fainberg [sic] (1982) does not directly provide such claims, only that such policies are ϵ-optimal.)

Kadota *et al.* (1994) discussed infinite horizon discounted problems with general continuous risk-sensitive utility functions for models with countable state spaces, more general action spaces, and nonnegative bounded rewards. They showed that there always exists an optimal policy with respect to any given initial state, and provided the optimality equations. Kadota *et al.* (1998a) further discussed stopping problems under the same objective, where the agent can choose to reach a goal state deterministically from any state, and provided a sufficient condition for an optimal policy. Kadota *et al.* (1998b) extended their previous work on stopping problems to total undiscounted rewards.

[7] Eugene Feinberg's last name was initially spelt as Fainberg due to a transcription from Russia.

One important approach is to augment the state space with the accumulated rewards. This approach converts a risk-sensitive planning problem to a risk-neutral planning problem, for which a rich literature is available. White (1987) used the state augmentation approach to discuss maximizing expected general risk-sensitive utility of total discounted rewards. Kerr (1999) used the state augmentation approach to solve a finite horizon dynamic programming problem with general risk-sensitive utility functions where the rewards are not necessarily additive. Müller (2000) used the state augmentation approach to discuss optimal stopping problems with general risk-sensitive utility functions.

The only other work dealing with general risk-sensitive utility functions from the DT planning community is (Dolgov and Durfee, 2004). This work proposed to use a series of moments of the total (undiscounted) reward to approximate the expected utility, and to use Legendre polynomials to approximate the cumulative distribution function of the total reward so that these moments can be calculated in closed forms. This work has several problems, however. First, this method can only be used to calculate moments for stationary policies, which we know are not sufficient for the MEU objective with general risk-sensitive utility functions (see above and Chapter 4). Second, this work actually only proposed a method for approximating the expected utilities for a given stationary policy, not a method for obtaining an optimal policy nor even a method for obtaining a policy that is optimal in the class of stationary policies. Third, this method requires that the total rewards are bounded,[8] which greatly limits its applicability.

2.4.4 MDPs with Even More General Utility Functions

The more general utility function for MDPs is a utility function defined directly on the set of trajectories, $\vec{U}(h)$. Utility functions defined in this way are interesting from the first principles of utility theory and Markov models, but they are not practical without further structural assumptions about the utility function or the model. Therefore, works related

[8] They argued that the method can still be used if the total rewards are bounded with a probability close to one. However, they cannot quantify this probability, and we know that risk-sensitive planning is sensitive to events with low probabilities.

to such utility functions are inherently limited to discussions such as the existence conditions of optimal values, the existence of optimal policies within special classes of policies, and the required properties of optimal policies (Kreps, 1977a,b, 1978; Fainberg [sic], 1982; Schäl, 1981). On the other hand, computational procedures are impossible for such general definition of utility functions, or at best only of theoretical significance.

In a series of papers (Kreps, 1977a,b, 1978), Kreps discussed maximizing expected utility of trajectories within the set of HD policies for countable state space models. Kreps (1977a) discussed upper and lower convergent problems, and Kreps (1978) discussed more general upper and lower transient problems, all of which can be viewed as different generalizations of the negative and positive models, respectively. For these problems, the optimal values exist but can be infinite. Kreps (1977b) considered problems with summary states, where a summary state summarizes the history up to the current decision epoch and the utility function defined on the complete trajectory is required to be equivalent to the utility function defined on the current summary state and the same tail of the given trajectory. Kreps (1977b) also showed that under the upper/lower or convergent/transient conditions, along with some technical conditions, a "stationary" deterministic policy exists, where the stationarity is with respect to the summary states.

Fainberg [sic] (1982) showed relations among different classes of policies for some even more general planning objectives, which is based on a direct mapping from a policy to a numerical value. When specialized to the case of maximizing expected utility of trajectories, Fainberg [sic] (1982) showed that there exists an MD (HD) policy that approximates any given MR (HR) policy, but there is no guarantee that the optimal values exist or are finite.

Hill and Pestien (1987) showed that for the objective of maximizing expected utility of complete trajectories, the optimal values can be obtained by considering only HD policies if the utility function is bounded, thus partially completing Kreps's results, which are obtained only within the set of HD policies.

For the same planning objective and models with more general state and action spaces, Schäl (1981) gave more conditions under which a policy is optimal, and showed the equivalence of some of such conditions if an optimal policy exists, but he did not answer the question under what conditions the optimal values and optimal policies exist.

2.5 Related Work: Large-Scale MDPs under Risk-Neutral Objectives

In this section, we review recent developments in DT planning for solving large-scale MDPs under risk-neutral objectives. These developments provide ideas that serve as the basis for our construction of solution methods for planning problems under risk-sensitive planning objectives, especially those using exponential utility functions.

Solution methods for large-scale problems under risk-neutral objectives can be roughly classified as using a symbolic strategy or a numerical strategy. In this thesis, I consider reusing ideas from methods with a symbolic strategy, but methods from both categories will be reviewed in this section.

2.5.1 Symbolic Strategies

Common symbolic strategies in AI include ideas such as search, temporal abstraction, and state abstraction. All of these strategies have been used to solve large-scale MDPs. Search-based methods explore only the relevant part of the state space to avoid enumerating all states. However, search-based methods do not guarantee that only a small number of states are visited and the problem may still be intractible. Therefore, in general, abstractions are needed to deal with large-scale problems. Temporal and state (or spatial) abstractions are two orthogonal abstraction strategies, where temporal abstraction uses “long-term” abstract actions and state abstraction uses abstract states that correspond to multiple “real” states.

2.5.1.1 Search

Search-based methods deal with the problem of reaching a set of goal states from a given set of initial states. The benefit of search methods is that when solving the problem, we only need to care about states that can be reached in the process of solving the problem.

Therefore, search methods provide partial policies that do not specify the actions for those states that need not be reached. Another benefit of search methods is that they can use heuristics, which are rough estimates of the true value functions based on prior knowledge, to speed up the convergence.

The real-time dynamic programming (RTDP) method (Barto *et al.*, 1995) is an online search method. It can be viewed as a stochastic generalization of LRTA*, which is a real-time search method for deterministic problems (Korf, 1990). The RTDP method performs an asynchronized version of value iteration based on simulation. In each step of the search, RTDP only updates the value of the current state the agent is in, and the agent commits to the currently-known best action right away. RTDP has been extended to solve large-scale MDPs in (Bonet and Geffner, 2002, 2003b,a; Feng *et al.*, 2003). The resulting policies from these methods may not deal with all contingencies, but only contingencies with sufficiently high probabilities.

LAO* (Hansen and Zilberstein, 2001) is an offline search method, as a generalization of the AO* method for search AND-OR graphs (Nilsson, 1980). MDPs can be viewed as AND-OR graphs, where the states are OR nodes and the actions are AND nodes. For an OR node, the solution depends on exactly one child, which is the max operation over all actions. For an AND node, the solution depends on all children, which is the expectation operation over all possible outcomes. The major difference between LAO* and AO* is that LAO* can deal with loops. LAO* can be seen as interleaving search and dynamic programming, where search is guided by the values obtained by dynamic programming and dynamic programming is only performed on a subset of states obtained by search, which are part of the best solution currently known. LAO* has also been extended to deal with problems represented in a factored form (see Section 2.5.1.3) in (Feng and Hansen, 2002; Hansen *et al.*, 2002). Different from RTDP and its descendants, LAO* and its descendants deal with all possible contingencies. LAO* and related methods will be discussed in more detail in Section 3.5, when they are used as examples for our transformation-of-algorithms approach.

2.5.1.2 Temporal Abstraction

Temporal abstraction are used to overcome the problem that (primitive) actions, as provided in the problem formulation, are often too fine-grained for solving large-scale planning problems. Therefore, temporal abstraction in AI planning uses abstract actions, or options following (Parr, 1998; Precup, 2000), to express "long-term" effects, where options are canned policies for parts of the state space to achieve some subtasks. For example, the robot navigation problem from Section 1.2 can be solved using options that can be interpreted as: "traverse the top path", "make the top right turn", "traverse the downward path", "make the bottom right turn", and "traverse the bottom path". Temporal abstraction methods are originally developed as reinforcement learning methods (Sutton *et al.*, 1999b; Parr and Russell, 1998; Dietterich, 2000), but the ideas have also been used in DT planning methods (Hauskrecht *et al.*, 1998).

Parr (1998) and Precup (2000) showed that options can be viewed as regular actions, and the original MDP can be reduced to an MDP (if discounting is not used) or a semi-MDP (if discounting is used) that uses only options. Hauskrecht *et al.* (1998) provided a planning method that uses a special type of options.

The MAXQ (Dietterich, 2000) and HAM (Parr and Russell, 1998; Parr, 1998) methods are hierarchical methods, which use task decomposition and incompletely specified policies. Subtasks (in MAXQ) and abstract machines (in HAM) are reduced to options in their respective solution processes. For this reason, the terms hierarchical methods and temporal abstraction are often used interchangeably in the DT planning and reinforcement learning literature.

Current methods using temporal abstractions assume that the abstraction or hierarchy is given to the planner. Such an abstraction or a hierarchy expresses domain knowledge that can be used to speed up planning. This is in line with hierarchical planning in classical AI planning research (Tate, 1977; Nau *et al.*, 1999). Options and hierarchical methods will be discussed in more detail in Chapter 3, as the basis for the pseudo-probability variant of our transformation-of-algorithms approach.

2.5.1.3 State Abstraction

State abstraction is used to overcome the problem that states, as provided in the problem formulation, are also often too fine-grained for solving large-scale problems. Therefore, state abstraction in AI planning uses abstract states to express "wide-range" properties, where an abstract state corresponds to multiple states in the original problem formulation. For this reason, some state abstraction methods are also referred to as state aggregation methods.

A convenient representation for state abstraction uses factored MDPs (Boutilier *et al.*, 1999). A factored MDP is a feature-based representation. In a factored MDP, the state is represented as a tuple of values of all features and the state space is the cross product of the sets of values of all features. Moreover, the transition probabilities for actions are represented compactly using 2-stage temporal Bayesian networks (Dean and Kanazawa, 1989; Boutilier *et al.*, 1999). More details of the factored representation are given in Section 3.7.1.

The SVI (Boutilier *et al.*, 1995, 2000) and SPUDD (Hoey *et al.*, 1999) methods use decision trees and algebraic decision diagrams, respectively, to represent the transition probabilities of factored MDPs even more compactly, by grouping together entries of the same values. Consequently, the value functions can also be represented compactly using decision trees or algebraic decision diagrams, where states with the same value are grouped together as an abstract state. SVI and SPUDD can be generalized to perform approximate dynamic programming where states with similar values are grouped together (Boutilier and Dearden, 1996; St. Aubin *et al.*, 2000). The SPUDD method has also been combined with search methods to solve even larger problems (Feng and Hansen, 2002; Hansen *et al.*, 2002; Feng *et al.*, 2003). We discuss SPUDD and related methods in more detail in Chapter 3, as an example of the pseudo-discount factor variant of our transformation-of-algorithms approach.

Givan *et al.* (2003) used model minimization as a general framework for state abstraction. This approach tries to find the "smallest" model that is equivalent to the original MDP. In particular, solution methods for factored MDPs (Boutilier *et al.*, 2000; Hoey *et al.*, 1999) can be viewed as special cases of model minimization. Model minimization can also be done approximately, and the resulting models are called bounded parameter MDPs, which have imprecise transition probabilities and reward functions. Such models can be solved using the method from (Givan *et al.*, 2000).

2.5.2 Numerical Strategies

Numerical strategies originated in numerical analysis and operations research, but are increasingly accepted by AI researchers. For DT planning, numerical strategies include methods using function approximators, direct policy search, and samping-based methods. We only briefly review DT planning methods using a numerical strategy.

2.5.2.1 Function Approximators

Function approximators have been used to approximate the values of states or the values of state-action pairs. Example function approximators include CMACs, radial basis functions, SDMs, and neutral networks (Sutton and Barto, 1998; Bertsekas and Tsitsiklis, 1996).

For factored MDPs, a class of promising function approximators are linear function approximators, which are used to approximate the value functions. A linear function approximator is a linear combination of a fixed number of basis functions, each of which only depend on a small number of factors. Methods from (Koller and Parr, 1999, 2000; Guestrin *et al.*, 2001, 2003; Poupart *et al.*, 2002; Patrascu *et al.*, 2002) are based on linear function approximators.

2.5.2.2 Direct Policy Search

Another numerical strategy for solving large-scale MDPs is to directly search for an "optimal" policy in a class of policies, which are parameterized. The "optimal" policy can only be optimal in the class of parameterized policies where the search takes place. Methods like those of (Baxter and Bartlett, 2001; Baxter *et al.*, 2001) and (Sutton *et al.*, 1999a) perform a gradient search in a subspace of the class of SR policies Π^{SR}, which is a continuous space. They both represent the SR policy using a parameterized representation, but estimate the gradients differently.

2.5.2.3 Sampling-Based Methods

Sampling-based methods use the planning problem specification as a generative model to generate samples. In fact, model-free reinforcement learning methods can all be used in this fashion and thus viewed as sampling-based methods (Sutton and Barto, 1998, Chapter 9). Kearns *et al.* (2002) describes a "pure" sampling-based method. Different from reinforcement learning methods, their method has explicit bounds on the number of samples needed to approximate the optimal values within a given error, but the sample complexity is exponential in the look-ahead. Sampling-based methods are also used to evaluate given policies. Ng and Jordan (2000) is a different sampling-based method and the sample complexity is polynomial for estimating a given policy. This method relies on policy search methods for optimization. Sampling is also used in (Fern *et al.*, 2003) as the policy evaluation method in an approximate policy iteration procedure.

CHAPTER III

EXPONENTIAL UTILITY FUNCTIONS

Exponential utility functions are the most widely used risk-sensitive utility functions (Corner and Corner, 1995) because they can model a spectrum of risk attitudes. We first define exponential utility functions and discuss their properties in Section 3.1, and then review in Section 3.2 basic results for solving MDPs under the MEU objective with exponential utility functions.

To solve large-scale planning problems under risk-sensitive planning objectives with exponential utility functions, I propose that existing decision-theoretic planners using a symbolic strategy can be transformed to take risk attitudes into account, as illustrated in Figure 3.1. The transformed algorithms bear visual resemblance to the original algorithms but special conditions are needed to ensure their validity. The transformation-of-algorithms approach is more general than the previous representation transformation approach (Figure 3.2), which transforms the planning tasks (Koenig and Simmons, 1994a,b; Koenig, 1997). In order to understand the advantages and limitations of the representation transformation approach, a detailed analysis is provided in Section 3.3. Next, Section 3.4 discusses the transformation-of-algorithms approach, and introduces the two transformation: one with pseudo-probabilities and one with transition-dependent pseudo-discount factors. If the probabilities are given explicitly, Section 3.5 shows that these two transformations are almost the same, using the LAO* method as an example. The LAO* method is also used to demonstrate how the search strategy can be used when solving risk-sensitive planning problems. The two transformations differ when the probabilities are given implicitly: the pseudo-probability transformation is more convenient for temporally extended probabilities, which result from temporal abstraction; and the pseudo-discount factor transformation is more convenient for factored probabilities, which result from state abstraction. They are

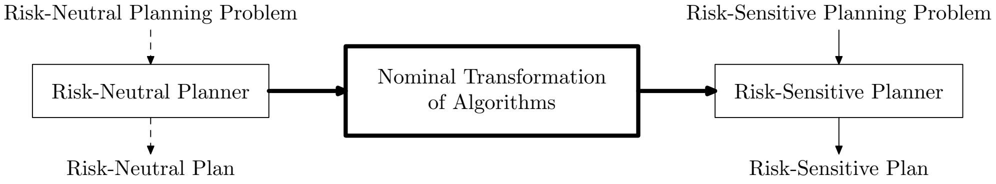

Risk-Neutral Planning Problem
Risk-Neutral Planner
Risk-Neutral Plan
Nominal Transformation of Algorithms
Risk-Sensitive Planning Problem
Risk-Sensitive Planner
Risk-Sensitive Plan

Figure 3.1: The "nominal transformation of algorithms" approach

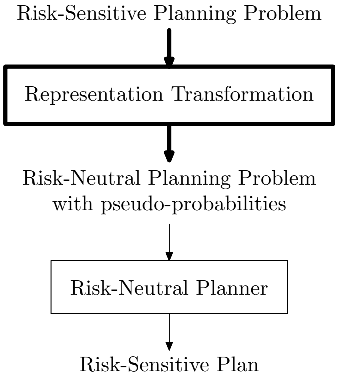

Risk-Sensitive Planning Problem
Representation Transformation
Risk-Neutral Planning Problem with pseudo-probabilities
Risk-Neutral Planner
Risk-Sensitive Plan

Figure 3.2: The "representation transformation" approach

discussed in detail using DT planning methods with temporal and state abstraction strategies in Section 3.6 and Section 3.7, respectively. Since many DT planning methods use one of these two types of implicitly represented probabilities and thus one of these two abstraction strategies, we anticipate that our transformation-of-algorithms approach can be widely applied.

Following the convention in Table 2.1, we use subscripts $_{\exp}$ in place of $_U$ to indicate that an exponential utility function is used and, if the risk parameter needs to be specified explicitly, we use subscripts $_{\exp(\gamma)}$.

3.1 Exponential Utility Functions

Exponential utility functions are the most widely used risk-sensitive utility functions. In a review paper, Corner and Corner (1995) noted that 27.5% of the applications they reviewed adopt exponential utility functions, while two thirds of them just use expected values and are thus risk-neutral.

Exponential utility functions have the form

$$U_{\exp}(w) = \begin{cases} \gamma^w, & \gamma > 1 \\ \\ -\gamma^w, & 0 < \gamma < 1 \end{cases}$$

To simplify the notation, let $\iota = \operatorname{sgn} \ln \gamma$, where sgn is the sign function. Then

$$U_{\exp}(w) = \iota \gamma^w, \tag{3.1}$$

and its inverse is

$$U_{\exp}^{-1}(u) = \log_\gamma(\iota u). \tag{3.2}$$

Exponential utility functions are constantly risk-sensitive utility functions, since the risk measure for an exponential utility function is a constant, which can be verified as follows. We have

$$U'_{\exp}(w) = \iota \log \gamma \gamma^w, \qquad U''_{\exp}(w) = \iota (\log \gamma)^2 \gamma^w,$$

and according to Eq. (2.2), the risk measure is

$$R_{\exp}(w) = -\frac{U''_{\exp}(w)}{U'_{\exp}(w)} = -\frac{\iota (\log \gamma)^2 \gamma^w}{\iota \log \gamma \gamma^w} \equiv -\log \gamma.$$

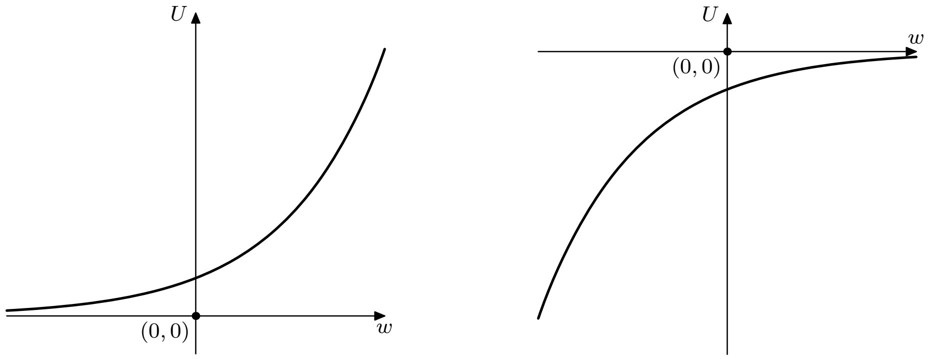
When $\gamma > 1$, the utility function is convex and $R_{\exp} < 0$, indicating the agent is risk-seeking (Figure 3.3(a)). When $0 < \gamma < 1$, the utility function is concave and $R_{\exp} > 0$, indicating the agent is risk-averse (Figure 3.3(b)). Being constantly risk-sensitive also implies that the preferences do not change with the wealth level (Pratt, 1964).

For a given lottery involving a finite number of wealth levels as outcomes, $\mathbf{w} = [w_1, p_1; w_2, p_2; \cdots, w_n, p_n]$, it is easy to verify the following results (Koenig and Simmons, 1994a):

$$\lim_{\gamma \downarrow 0} \mathrm{ce}_{\exp}(\mathbf{w}) = \min_{1 \le i \le n} w_i,$$

$$\lim_{\gamma \to 1} \mathrm{ce}_{\exp}(\mathbf{w}) = \sum_{i=1}^{n} p_i w_i = E[\mathbf{w}],$$

$$\lim_{\gamma \to +\infty} \mathrm{ce}_{\exp}(\mathbf{w}) = \max_{1 \le i \le n} w_i.$$



(a) Convex (Risk-Seeking)

(b) Concave (Risk-Averse)

Figure 3.3: Exponential utility functions

This result means that by varying the parameter γ, we can make different tradeoffs among the best-case, expected-case, and worst-case scenarios. Therefore, the exponential utility function can model a spectrum of (constant) risk attitudes.

3.2 An Overview of Basic Properties

We start with a brief overview of the basic properties for planning with MDP models under the $\mathsf{MEU}_{\mathrm{exp}}$ planning objective. We only present results that are relevant to our transformation-of-algorithms approach.

Similar to the MER objective, the $\mathsf{MEU}_{\mathrm{exp}}$ objective is also decomposable, since for any random variables $\mathbf{x}$ and $\mathbf{y}$, it holds that

$$E\big[U_{\mathrm{exp}}(\mathbf{x}+\mathbf{y})\big] = E\left[\iota\gamma^{\mathbf{x}+\mathbf{y}}\right] = E\big[\gamma^{\mathbf{x}}E\left[\iota\gamma^{\mathbf{y}}\right]\big] = E\Big[\gamma^{\mathbf{x}}E\big[U_{\mathrm{exp}}(\mathbf{y})\big]\Big]. \qquad (3.3)$$

However, we cannot decompose it further to be $E[\gamma^{\mathbf{x}}]E\big[U_{\mathrm{exp}}(\mathbf{y})\big]$ if $\mathbf{x}$ and $\mathbf{y}$ are not independent, which in general is the case for the rewards received in MDP models. Therefore, the decomposition is a kind of "partial" decomposition, in contrast to the "complete" decomposition for linear utility functions discussed in Section 2.3. In fact, this property requires us to use the reward model $r(s,a,s')$ rather than the reward model $r(s,a)$ (Chung and Sobel, 1987). The reason is as follows. Consider state s and action $a \in A_s$. For any random total future reward $\mathbf{r}$ that is dependent on next-time state s', we have

$$E^{s,a}\big[U_{\mathrm{exp}}(r_0+\mathbf{r})\big] = E_{s'}\Big[E^{s,a}\big[U_{\mathrm{exp}}(r_0+\mathbf{r})\,\big|s'\big]\;\Big] = E_{s'}\Big[E^{s,a}\Big[\gamma^{r_0}E\big[U_{\mathrm{exp}}(\mathbf{r})\,\big|s'\;\big]\Big|s'\Big]\Big]$$

$$= E_{s'}\left[E^{s,a}\left[\gamma^{r_0} E^{s'}\left[U_{\exp}(\mathbf{r})\right]\middle|s'\right]\right]$$

$$= E_{s'}\left[E^{s,a}\left[\gamma^{r_0}\,\middle|s'\,\right]E^{s'}\left[U_{\exp}(\mathbf{r})\right]\right],$$

where the last equality holds since r_0 and $\mathbf{r}$ are independent given next-time state s'. Comparing to Eq. (3.3), we can see that an appropriate reward model needs to involve current-time state s, action a, and next-time state s', that is, $\gamma^{r(s,a,s')} = E^{s,a}\left[\gamma^{r_0}\,|s'\right]$, or $r(s,a,s') = \log_\gamma E^{s,a}\left[\gamma^{r_0}\,|s'\right]$ for the $\mathsf{MEU}_{\exp}$ planning objective. If we consider the most general reward model where the immediate reward received for a transition (s,a,s') forms a distribution $F_r(r|s,a,s')$ (see Section 2.3), we have

$$r(s,a,s') = \log_\gamma E^{s,a}\left[\gamma^{r_0}\,\middle|s'\right] = \log_\gamma \int \gamma^r F_r(dr|s,a,s').$$

We consider finite MDP models under the $\mathsf{MEU}_{\exp}$ planning objective. Similar to the case of the MER objective, we define the value of a state $s \in S$ under a policy π as the limit of the finite-horizon values as T approaches infinity

$$v_{\exp}(s) = \lim_{T\to\infty} v_{\exp,T}(s) = \lim_{T\to\infty} E\left[U_{\exp}\left(\sum_{t=0}^{T-1} r_t\right)\right] = \lim_{T\to\infty} E\left[\iota\gamma^{w_T}\right].$$

The value $v_{\exp}(s)$ may not exist, or may be infinite. So we need to specify conditions under which such values exist and are finite. In this chapter, we only consider models with either nonpositive or nonnegative rewards. Therefore, it is more convenient to first consider problems without explicit goal states in Section 3.2.1. Then we consider additional results for problems with goal states in Section 3.2.2. These results resemble their counterparts under the MER objective, and thus are the basis for our discussion of the transformation-of-algorithm approach.

3.2.1 Problems without Explicit Goal States

Assuming that Condition 2.1 (Finite Model) holds and the expected utility of total rewards exist under all policies, Ávila-Godoy (1999) showed that the optimal values satisfy the system of optimality equations

$$v_{\exp}(s) = \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')} v_{\exp}(s'),$$

which does not have a unique solution, since for example, if $v^*_{\exp}(\cdot)$ is a solution, then $kv^*_{\exp}(\cdot)$ is also a solution for all $k \geq 0$.

In fact, most of the following results also hold under Condition 2.2 (Countable States), but we present a version under Condition 2.1 (Finite Model) since we focus on finite models in DT planning (see Section 1.5).

3.2.1.1 Negative Models

For negative models (Condition 2.5), similar to the MER case, the finite horizon values are monotonic in the planning horizon, and the values for all policies and the optimal values exist. The optimal values are finite if the following condition holds (Ávila-Godoy, 1999).

Condition 3.1 (Negative Model with Finite Exponential Certainty Equivalents)**.** There exists a policy $\pi \in \Pi$ such that for all states $s \in S$, the value $U^{-1}_{\exp}\left(v^{\pi}_{\exp}(s)\right)$ is finite.

Under Condition 2.1 (Finite Model), Condition 2.5, and Condition 3.1, the optimal values are the maximal solution to the optimality equations in the set of value functions that are no greater than $\iota = U_{\exp}(0)$ (Ávila-Godoy, 1999). Moreover, there exists an SD optimal policy (Ávila-Godoy, 1999).

A policy π is conserving under the $\mathsf{MEU}_{\exp}$ objective if for all states $s \in S$,

$$v^*_{\exp}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, \pi(s)) \gamma^{r(s,\pi(s),s')} v^*_{\exp}(s').$$

Ávila-Godoy (1999) also showed that a conserving policy is optimal.

Under Condition 2.1, Condition 2.5, and Condition 3.1, value iteration can be used to solve the MDP. The value iteration procedure uses the following value update rule

$$v^t_{\exp}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \gamma^{r(s,a,s')} v^{t-1}_{\exp}(s), \qquad s \in S,$$

and $v^t_{\exp}$ converges to the optimal values starting from an initial value function $v^0_{\exp}$ where for all states $s \in S$, $\iota \geq v^0_{\exp}(s) \geq v^*_{\exp}(s)$ (Ávila-Godoy, 1999). However, policy iteration cannot be used since a suboptimal value function can be a solution of the optimality equations, which is similar to the MER objective.

In fact, if $\gamma > 1$, the above results still hold if Condition 3.1 is violated, but it is possible that $v^*_{\exp}(s) = 0$ for some states s. This case and Condition 3.1 can be summarized in the following condition.

Condition 3.2 (Negative Model with Finite Exponential Utilities)**.** There exists a policy $\pi \in \Pi$ such that for all states $s \in S$, the value $v^{\pi}_{\exp}(s)$ is finite.

3.2.1.2 Positive Models

Similar to negative models, for positive models (Condition 2.7), the finite horizon values are monotonic in the planning horizon, and the values for all policies and the optimal values exist. The optimal values are finite if Condition 2.1 and the following condition hold (Cavazos-Cadena and Montes-de-Oca, 2000a,b).

Condition 3.3 (Positive Model with Finite Exponential Certainty Equivalents)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the value $U^{-1}_{\exp}\left(v^{\pi}_{\exp}(s)\right)$ is finite.

Under Condition 2.1 (Finite Model), Condition 2.7, and Condition 3.3, the optimal values are the minimal solution to the optimality equations in the set of value functions that are no less than $\iota = U_{\exp}(0)$ (Cavazos-Cadena and Montes-de-Oca, 2000a). There also exists an SD-optimal policy (Cavazos-Cadena and Montes-de-Oca, 2000a).

Under Condition 2.1, Condition 2.7, and Condition 3.3, Cavazos-Cadena and Montes-de-Oca (2000b) showed that a conserving SD policy that is also unichain is optimal. A stationary policy is unichain if and only if the induced Markov chain has only a single recurrent class.[1]

Under Condition 2.1, Condition 2.7, and Condition 3.3, we can use value iteration and policy iteration. Under these conditions, the value iteration procedure converges to the optimal values starting from an initial value function $v^0_{\exp}$ such that for all states $s \in S$, $\iota \le v^0_{\exp}(s) \le v^*_{\exp}(s)$ (Cavazos-Cadena and Montes-de-Oca, 2000b). The value update rule is the same as the one for negative models.

[1] A recurrent class of a Markov chain is a set of states that are recurrent and reachable from each other.

The policy iteration procedure can also be used to obtain an SD-optimal policy, where the policy evaluation equation is

$$v_{\exp}(s) = \sum_{s' \in S} P(s'|s, \pi(s)) \gamma^{r(s,\pi(s),s')} v_{\exp}(s'), \qquad s \in S.$$

It is, however, necessary to set the values for recurrent states under π to $\iota = U_{\exp}(0)$, since the solution is not unique (Cavazos-Cadena and Montes-de-Oca, 2000b).

Similar to negative models, if $0 < \gamma < 1$, the results also hold without Condition 3.3, and it is possible that $v^*_{\exp}(s) = 0$ for some states s. This case and Condition 3.3 can be summarized as the following condition.

Condition 3.4 (Positive Model with Finite Exponential Utilities)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the value $v^\pi_{\exp}(s)$ is finite.

3.2.2 Problems with Explicit Goal States

For a problem with goal states, the value of a goal state is always $U_{\exp}(0) = \iota$ for any policy. If we assume the optimal values exist, we have the following system of optimality equations

$$\begin{aligned} v_{\exp}(s) &= \iota, && s \in G, \\ v_{\exp}(s) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \gamma^{r(s,a,s')} v_{\exp}(s'), && s \in S \setminus G. \end{aligned}$$

It is natural to assume that Condition 2.3 (Proper Policy) holds for problems with goal states.

3.2.2.1 Negative Models

We consider finite negative models with goal states in this section. From Section 3.2.1.1, we know that there exists an SD-optimal policy, and a conserving policy is optimal. Moreover, the optimality equations have the optimal values as the unique solution under Condition 3.1 (Denardo and Rothblum, 1979).

A strictly negative model is a negative model with goal states whose rewards are strictly negative before a goal state is reached.

Condition 3.5 (Strictly Negative Model)**.** For all $s \in S \setminus G$ ($G \neq \varnothing$), all $a \in A_s$, and all $s' \in S$ where $P(s'|s, a) > 0$, it holds that $r(s, a, s') < 0$.

For risk-averse agents and strictly negative models, Condition 3.1 (or Condition 3.2) implies Condition 2.3 (Proper Policy) (Patek, 2001). Under the same conditions, an SD-optimal policy is also proper (Patek, 2001). In this case, both value iteration and policy iteration can be used to solve such problems, where value iteration can start with any negative initial values, and policy iteration needs to start with a proper SD policy with finite values (Patek, 2001).

For risk-seeking agents, the values are bounded under all policies. An SD-optimal policy is proper only if for all states $s \in S$, the optimal values $v^*_{\text{exp}}(s)$ are positive (Denardo and Rothblum, 1979), that is, if Condition 3.1 holds. Both value iteration and policy iteration can be used to solve such problems, where value iteration can start with any positive initial values, and policy iteration needs to start with a proper SD policy, which always has finite values. In fact, the correctness of value iteration and policy iteration can be obtained by using the representation transformation (Koenig and Simmons, 1994a; Koenig, 1997, see also Section 3.3).

3.2.2.2 Positive Models

We consider finite positive models with goal states in this section. From Section 3.2.1.2, we know that there exists an SD-optimal policy, and that value iteration and policy iteration procedures can be used to solve such problems under Condition 3.4. Moreover, under Condition 3.3, the optimality equations have the optimal values as the unique solution, an SD-optimal policy is proper, and a conserving proper policy is optimal (Denardo and Rothblum, 1979).

Under Condition 3.3, value iteration can start with any negative (positive) initial values for risk-averse (risk-seeking) agents, and policy iteration needs to start with a proper SD policy and keeps all improved policies proper to obtain a proper SD-optimal policy.

3.2.3 Online Testing of Finiteness Conditions

It is important to know whether the optimal value function is finite, since otherwise value iteration and policy iteration may not produce meaningful results, or may not even terminate at all. For problems with goal states, one way to test the finiteness conditions is to use the linear programming formulation of (Denardo and Rothblum, 1979), which in fact also solves the MDP. However, most of DT planning methods take the dynamic programming approach and are based on value iteration or policy iteration. It is therefore desirable to be able to test the conditions online as value iteration or policy iteration proceeds. In this section, we show that online testing can be done for problems with goal states.

The values are always finite if the model is positive and the agent is risk-averse, or if the model is negative and the agent is risk-seeking. We consider the other two cases with goal states, which are also the most interesting for risk-sensitive planning: risk-averse agents and strictly negative models, as well as risk-seeking agents and positive models. We also assume that Condition 2.3 (Proper Policy) holds. Under these settings, there exists a proper SD-optimal policy.

3.2.3.1 Equivalent Conditions for Proper Policies with Infinite Values

We now consider some conditions equivalent to policies with infinite values. These conditions will be tested online. Notice that in the two settings we consider, it holds that $\gamma^{r(s,a,s')} \geq 1$ for all valid transitions (s, a, s').

First, the finiteness property of a proper SD policy π can be determined by solving the system of policy evaluation equations for π:

$$\begin{aligned} v_{\exp}(s) &= \iota, && s \in G, \\ v_{\exp}(s) &= \sum_{s' \in S} P(s'|s, \pi(s)) \gamma^{r(s,\pi(s),s')} v_{\exp}(s'), && s \in S \setminus G. \end{aligned}$$

The value $v^{\pi}_{\exp}(s)$ is finite for all states $s \in S$ if and only if the above system has a unique solution whose elements have the same sign as ι (Patek, 2001). Therefore if $v^{\pi}_{\exp}(s)$ is infinite for some states, then the system has no solution, multiple solutions, or an invalid solution. A solution $v_{\exp}$ is invalid if $\iota = 1$ but $v_{\exp}(s) \leq 0$, or if $\iota = -1$ but $v_{\exp}(s) \geq 0$. They are invalid since the value functions should be nonnegative (nonpositive) if $\iota = 1$ ($\iota = -1$). These situations can be illustrated using a simple example shown in Figure 3.4, where the double circle indicates a goal state. There is only one policy π. We have the following system of policy evaluation equations

$$\begin{aligned}
v_{\exp}(s^1) &= \frac{1-p}{2}\gamma^r v_{\exp}(s^1) + \frac{p}{2}\gamma^r v_{\exp}(s^2) + \frac{1}{2}\gamma^r v_{\exp}(s^3),\\
v_{\exp}(s^2) &= \frac{p}{2}\gamma^r v_{\exp}(s^1) + \frac{1-p}{2}\gamma^r v_{\exp}(s^2) + \frac{1}{2}\gamma^r v_{\exp}(s^3),\\
v_{\exp}(s^3) &= \iota.
\end{aligned}$$

The coefficient matrix of the above system is

$$\begin{pmatrix} \frac{1-p}{2}\gamma^r - 1 & \frac{p}{2}\gamma^r & \frac{1}{2}\gamma^r \\ \frac{p}{2}\gamma^r & \frac{1-p}{2}\gamma^r - 1 & \frac{1}{2}\gamma^r \\ 0 & 0 & 1 \end{pmatrix},$$

and its determinant is $\left(\frac{1}{2}\gamma^r - 1\right)\left(\frac{1}{2} - p - \gamma^{-r}\right)\gamma^r$. We notice that if $0 \le p = \frac{1}{2} - \gamma^{-r} \le 1$, then the first two equations reduce to the same form, and thus the system has an infinite number of solutions. We also notice that if $\gamma^r = 2$, then the first two equations are inconsistent, and thus the system has no solution. In other cases, the system has a solution

$$v_{\exp}(s^1) = v_{\exp}(s^2) = \frac{\gamma^r \iota}{2 - \gamma^r}, \qquad v_{\exp}(s^3) = \iota.$$

But if $\gamma^2 > 2$ (and $p \neq \frac{1}{2} - \gamma^{-r}$), $v_{\exp}(s^1)$ and $v_{\exp}(s^2)$ have a sign different from $v_{\exp}(s^3)$, and thus this solution is invalid in this case. On the other hand, it is easy to verify that the values $v_{\exp}(s^1)$ and $v_{\exp}(s^2)$ are finite only if $\gamma^r < 2$ and infinite otherwise.

Although we can determine whether a proper SD policy has infinite values by solving the policy evaluation equations, it is an expensive operation. It is desirable to use alternatives that are computationally more efficient. We notice that the finiteness property of a proper SD policy is related to the spectral radius of a matrix. For a given SD policy π, the matrix D_π is defined such that its (s, s') entry is

$$D_\pi(s, s') = P(s'|s, \pi(s))\gamma^{r(s,\pi(s),s')}, \qquad s, s' \in S. \tag{3.4}$$

Then the T-horizon value for state s is

$$v^\pi_{\exp,T}(s) = \iota \sum_{s' \in S} D^T_\pi(s, s'), \tag{3.5}$$

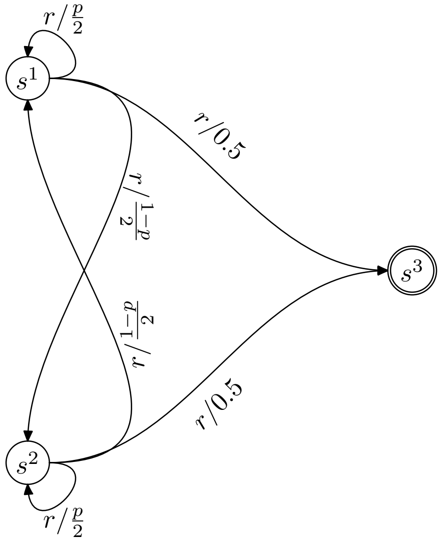


Figure 3.4: An example MDP with infinite values

where D_π^T is the T-th power of D_π (Patek, 2001). If $G \neq \varnothing$, the definition of D_π can be refined as

$$D_\pi(s,s') = \begin{cases} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')}, & s \in S \setminus G, \\ 1, & s = s', s \in G, \\ 0, & s \neq s', s \in G, \end{cases}$$

since $r(s,\pi(s),s') = 0$ and $P(s|s,\pi(s)) = 1$ for all $s \in G$. Now suppose π is proper. We can order the states so that the last $|G|$ rows and columns of D_π correspond to goal states, that is,

$$D_\pi = \begin{pmatrix} A_\pi & B_\pi \\ \mathbf{0} & \mathbf{1} \end{pmatrix}. \tag{3.6}$$

The spectral radius $\rho(A_\pi)$ is the largest magnitude of A_π's eigenvalues. The spectral radius $\rho(A_\pi) \geq 1$ if and only if the values of some states for π are infinite (Patek, 2001, see also Section 5.3). The spectral radius of a nonnegative matrix can be calculated more efficiently than solving the system of policy evaluation equations (Jia, 1998).

An SD policy has infinite values because the agent has a possibility of looping in a part of the state space. Identifying the part of the state space where the agent loops can rule out all SD policies that contain such a loop. Consider again the matrix D_π defined above.

We can arrange the states such that D_π has a canonical form (Seneta, 1981)

$$D_\pi = \begin{pmatrix} D_{11} & D_{12} & \cdots & D_{1k} & D_{1,k+1} & \cdots & D_{1\ell} \\ & D_{22} & \cdots & D_{2k} & D_{2,k+1} & \cdots & D_{2\ell} \\ & & \ddots & \vdots & \vdots & & \vdots \\ & & & D_{kk} & D_{k,k+1} & \cdots & D_{k\ell} \\ & & & & D_{k+1,k+1} & & \\ & & & & & \ddots & \\ & & & & & & D_{\ell\ell} \end{pmatrix}$$

where D_{ii} $(1 \leq i \leq \ell)$ are irreducible square matrices[2] along the main diagonal. Let S_i be the subset of states corresponding to block D_{ii}, and $S_i^+ = \bigcup\limits_{j=i}^{\ell} S_j$. It is known that the irreducible blocks correspond to strongly connected components of the graph $\mathcal{G}_\pi = (S, \mathcal{E}_\pi)$ (Seneta, 1981), where[3]

$$\mathcal{E}_\pi = \Big\{(s,s') \,\Big|\, s,s' \in S, P\big(s'|s,\pi(s)\big) > 0\Big\}.$$

The strongly connected components, and thus the irreducible blocks D_{ii}, can be found using a standard algorithm (Corman *et al.*, 1990). Notice that in particular, the blocks $D_{k+1,k+1}$ through $D_{\ell\ell}$ corresponds to recurrent states under π, and therefore $\ell \geq k+1$ since there exists at least one recurrent state.

The following theorem relates the finiteness of values for an SD policy π to the spectral radii of the irreducible blocks of D_π.

Theorem 3.1. *Assume that Condition 2.1 and Condition 2.5 (Condition 2.7) hold. Consider an SD policy π and the matrix D_π in the canonical form. Then the values $v^\pi_{\exp}(s)$ are finite for all states $s \in S$ if and only if for all irreducible blocks D_{ii} $(1 \leq i \leq \ell)$, either $\rho(D_{ii}) < 1$, or $\rho(D_{ii}) = 1$ and $r\big(s,\pi(s),s'\big) = 0$ for all states $s, s' \in S_i$.*

We will use the following result, which is a part of the famous Perron-Frobenius Theorem for finite nonnegative matrices.

[2] A matrix is irreducible if we cannot rearrange its rows and columns so that the nonzero elements have a block upper triangle form.

[3] We allow self-loops in $\mathcal{G}_\pi$.

Lemma 3.2 (Seneta 1981)**.** *If D_1 and D_2 are finite nonnegative matrices and D_1 is less than or equal to D_2 elementwise, denoted as $D_1 \leq D_2$, then $\rho(D_1) \leq \rho(D_2)$. If, in addition, D_2 is irreducible, $\rho(D_1) = \rho(D_2)$ implies that $D_1 = D_2$.* □

Proof of Theorem 3.1. We prove a stronger result by induction from ℓ to 1: for all $1 \leq i \leq \ell$, the values $v^{\pi}_{\text{exp}}(s)$ are finite for all state $s \in S^{+}_{i}$ if and only if for all irreducible blocks D_{jj} $(i \leq j \leq \ell)$, either $\rho(D_{jj}) < 1$ or $\rho(D_{jj}) = 1$ and $r\big(s, \pi(s), s'\big) = 0$ for all states $s, s' \in S_j$.

Let P_π be the transition probability matrix with the same structure as D_π, that is, $P_\pi(s, s') = P(s'|s, \pi(s))$ and

$$P_\pi = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1k} & P_{1,k+1} & \cdots & P_{1\ell} \\ & P_{22} & \cdots & P_{2k} & P_{2,k+1} & \cdots & P_{2\ell} \\ & & \ddots & \vdots & \vdots & & \vdots \\ & & & P_{kk} & P_{k,k+1} & \cdots & P_{k\ell} \\ & & & & P_{k+1,k+1} & & \\ & & & & & \ddots & \\ & & & & & & P_{\ell\ell} \end{pmatrix}$$

Notice that $\gamma^{r(s,a,s')} \geq 1$ under our assumptions. We thus have $P_\pi \leq D_\pi$ and $P_{ij} \leq D_{ij}$ for each block.

First we consider blocks corresponding to recurrent states, that is, the set S_i with $k+1 \leq i \leq \ell$. Since $P_{ii} \leq D_{ii}$, it holds that $\rho(D_{ii}) \geq 1$. If $r\big(s, \pi(s), s'\big) = 0$ for all transitions with $s, s' \in S_i$, then $D_{ii} = P_{ii}$ and thus $\rho(D_{ii}) = \rho(P_{ii}) = 1$. On the other hand, since only zero rewards are possible in this case, we have $v^{\pi}_{\text{exp}}(s) = \iota$ for all states $s \in S_i$. If $r\big(s, \pi(s), s'\big) \neq 0$ for some states $s, s' \in S_i$, the values for all states in S_i are infinite since this transition will be taken infinitely often. In this case $D_\pi \neq P_\pi$, and it must hold that $\rho(D_\pi) > 1$ due to Lemma 3.2. Therefore, the result holds for all $s \in S_i$ with $k+1 \leq i \leq \ell$.

Now we consider the set S_i with $1 \leq i \leq k$. Suppose that the result holds for states in S^{+}_{i+1}. We can assume that for all states $s \in S^{+}_{i+1}$, the value $v^{\pi}_{\text{exp}}(s)$ is finite. We define

$$\bar{D}_{\pi,i} = \begin{pmatrix} D_{ii} & D_{i,i+1} & \cdots & D_{i\ell} \\ & \mathbf{1} & & \\ & & \ddots & \\ & & & \mathbf{1} \end{pmatrix},$$

where the lower-right portion of D_π is replaced with an identity matrix. We can ignore the rows corresponding to states in $S \setminus S^{+}_{i}$ since the values of states in S_i cannot depend on those states.

We can consider a reduced MDP model that only includes states in S_i^+ with the same transition probabilities as the original model, and the same reward function except for transitions among states outside S_i, where the rewards are assumed to be zero. Therefore, $\rho(D_{ii}) \geq 1$ if and only if the values of some states in S_i^+ in the reduced model are infinite. In the reduced model, it is only possible that the value of a state in S_i is infinite. If the value of state $s \in S_i$ is infinite, the following sum approaches positive infinity as $T \to \infty$:

$$\sum_{s' \in S_i \cup \cdots \cup S_\ell} \bar{D}_{\pi,i}^T(s,s') = \sum_{s' \in S} \bar{D}_{\pi,i}^T(s,s').$$

Let

$$D_{\pi,i} = \begin{pmatrix} D_{ii} & \cdots & D_{ik} & D_{i,k+1} & \cdots & D_{i\ell} \\ & \ddots & \vdots & \vdots & & \vdots \\ & & D_{kk} & D_{k,k+1} & \cdots & D_{k\ell} \\ & & & D_{k+1,k+1} & & \\ & & & & \ddots & \\ & & & & & D_{\ell\ell} \end{pmatrix}$$

It holds $\bar{D}_{\pi,i} \leq D_{\pi,i}$ and thus $\bar{D}_{\pi,i}^T \leq D_{\pi,i}^T$. Then for a state $s \in S_i$, we have

$$\iota \cdot v_{\exp,T}^{\pi}(s) = \sum_{s' \in S} D_{\pi}^T(s,s') = \sum_{s' \in S} D_{\pi,i}^T(s,s') \geq \sum_{s' \in S_i} \bar{D}_{\pi,i}^T(s,s').$$

Therefore, if $\rho(D_{ii}) \geq 1$, the right hand side of the inequality and thus $\iota \cdot v_{\exp,T}^{\pi}(s)$ approaches positive infinity as $T \to \infty$. Moreover, if $\rho(D_{ii}) = 1$, there must be some $s, s' \in S_i$ such that $r\big(s, \pi(s), s'\big) \neq 0$. Otherwise D_{ii} will be a substochastic matrix since $D_{i,i+1}, \ldots, D_{i\ell}$ are not all zero, and thus $\rho(D_{ii}) < 1$, which is a contradiction.

Now suppose that $\rho(D_{ii}) < 1$. We consider the matrix

$$\hat{D}_{\pi,i} = \begin{pmatrix} D_{ii} & \cdots & D_{ik} & D_{i,k+1} & \cdots & D_{i\ell} \\ & \ddots & \vdots & \vdots & & \vdots \\ & & D_{kk} & D_{k,k+1} & \cdots & D_{k\ell} \\ & & & \mathbf{1} & & \\ & & & & \ddots & \\ & & & & & \mathbf{1} \end{pmatrix},$$

which replace the lower-right portion of $D_{\pi,i}$ with the identity matrix. Since we assume $v_{\exp}^{\pi}(s)$ is finite for all $s \in S_{i+1}^+$, we have $\rho(D_{jj}) < 1$ for $i+1 \leq j \leq k$ and $r\big(s, \pi(s), s'\big) = 0$ for all states

$s, s' \in S_k^+$. It holds that for all states $s \in S_i$ and all $T \in \mathbb{N}$

$$\iota \cdot v_{\exp,T}^{\pi}(s) = \sum_{s' \in S} D_{\pi}^{T}(s, s') = \sum_{s' \in S} D_{\pi,i}^{T}(s, s') = \sum_{s' \in S} \hat{D}_{\pi,i}^{T}(s, s').$$

Since we have

$$\rho \begin{pmatrix} D_{ii} & \cdots & D_{ik} \\ & \ddots & \vdots \\ & & D_{kk} \end{pmatrix} = \max_{i \le j \le k} \rho(D_{jj}) < 1,$$

the values of states in S_i^+ are all finite. Thus the result holds for all states in S. □

Let $\mathcal{H} = (S_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}})$ be a subgraph of the induced graph $\mathcal{G}_{\pi}$ where $S_{\mathcal{H}}$ is the set of vertices and $\mathcal{E}_{\mathcal{H}}$ is the set of edges. Define the square matrix $D_{\mathcal{H}}$ whose index set is $S_{\mathcal{H}}$ to be

$$D_{\mathcal{H}}(s, s') = \begin{cases} D_{\pi}(s, s'), & (s, s') \in \mathcal{E}_{\mathcal{H}}, \\ 0, & \text{otherwise}. \end{cases}$$

Lemma 3.3. *Assume that Condition 2.1 and Condition 2.5 (Condition 2.7) hold. Consider an SD policy π and the induced graph $\mathcal{G}_{\pi}$. Let $\mathcal{H} = (S_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}})$ be a strongly connected subgraph of $\mathcal{G}_{\pi}$ and $D_{\mathcal{H}}$ be the matrix corresponding to $\mathcal{H}$. If not all rewards corresponding to edges in $\mathcal{E}_{\mathcal{H}}$ are zero, then $\rho(D_{\mathcal{H}}) \ge 1$ implies that for all states $s \in S_{\mathcal{H}}$, the value $v_{\exp}^{\pi}(s)$ is infinite.*

Proof. Notice that $\mathcal{H}$ must be a subgraph of a strongly connected component of $\mathcal{G}_{\pi}$. Let this strongly connnected component be $\mathcal{G}_{\pi,i}$, the corresponding irreducible block be D_{ii}, and the corresponding set of states be S_i. Consider the matrix $\tilde{D}_{\mathcal{H}}$ of index set S_i where

$$\tilde{D}_{\mathcal{H}}(s, s') = \begin{cases} D_{\mathcal{H}}(s, s'), & (s, s') \in S_{\mathcal{H}}, \\ 0, & \text{otherwise}. \end{cases}$$

It holds that $\rho(D_{\mathcal{H}}) = \rho(\tilde{D}_{\mathcal{H}}) \ge 1$ and $\tilde{D}_{\mathcal{H}} \le D_{ii}$. Then according to Lemma 3.2, it holds that $\rho(D_{ii}) \ge 1$. The result follows from our assumption and Theorem 3.1. □

A partial SD policy π is a partial mapping from the state space S to the action space A. Let S_{π} be the set of states for which π is defined. Define $\mathcal{G}_{\pi} = (S_{\pi}, \mathcal{E}_{\pi})$ to be the induced subgraph of π where the vertex set is S_{π} and the edge set is

$$\mathcal{E}_{\pi} = \Big\{ (s, s') \,\Big|\, s, s' \in S_{\pi}, P\big(s' | s, \pi(s)\big) > 0 \Big\}.$$

Define a completion of π to be an SD policy that has the same mapping as π for states in S_π, denoted as π°. Then the following theorem follows directly from the above lemma.

Theorem 3.4. *Assume that Condition 2.1 and Condition 2.5 (Condition 2.7) hold. Consider a partial SD policy π and the induced graph $\mathcal{G}_\pi$. Let D_π be the matrix corresponding to $\mathcal{G}_\pi$. If $\mathcal{G}_\pi$ is strongly connected and not all rewards corresponding to edges in $\mathcal{E}_\pi$ are zero, then $\rho(D_\pi) \geq 1$ implies that for all states $s \in S_\pi$ and all completions of π (denoted π°), the value $v_{\text{exp}}^{\pi^\circ}(s)$ is infinite.* □

3.2.3.2 Positive Models and Condition 3.4

For positive models, one policy with infinite values is sufficient to violate Condition 3.4 (Positive Model with Finite Exponential Utilities).

Policy iteration should always start with a proper SD policy in our settings. Recall that policy iteration alternates between policy evaluation and policy improvement steps. Therefore, if any SD policy (including the initial policy) is evaluated to have infinite values, policy iteration needs to be terminated. Since policy evaluation is part of policy iteration, no additional tests are needed. Since there are only a finite number of SD policies, policy iteration will terminate.

We now consider value iteration. For positive models, there are two possibilities. If there is no proper greedy SD policy with respect to the current values, the optimal values are infinite since if a state is improper under the improper greedy SD policy (Test1), its value must be infinite. If, however, there always exists a proper greedy SD policy with respect to the current values, we can check the spectral radius $\rho(A_\pi)$ in the main loop and terminate value iteration if $\rho(A_\pi) \geq 1$ (Test2). Adding Test1 and Test2 does not change the value update rule, therefore the version of value iteration with these tests still converges to the optimal values if Condition 3.4 holds. Now we show that if Condition 3.4 does not hold, Test1 or Test2 will succeed within a finite time. Suppose Test1 and Test2 never succeed. Then all greedy policies are proper SD policies with finite values. Let the initial value function be v_{exp}^{0} and the greedy policy at iteration t be π^t. We have

$$v_{\text{exp}}^{t+1}(s) = \sum_{s' \in S} P(s'|s, \pi^t(s))\gamma^{r(s,\pi^t(s),s')} v_{\text{exp}}^{t}(s') = \sum_{s' \in S} D_{\pi_t}(s, s') v_{\text{exp}}^{t}(s'), \qquad s \in S \setminus G,$$

where D_{π^t} is the matrix corresponding to π^t as defined in Eq. (3.4). In a matrix form, it is

$$v_{\text{exp}}^{t+1} = D_{\pi^t} v_{\text{exp}}^{t}.$$

We therefore have

$$v_{\text{exp}}^{t+1} = D_{\pi^t} D_{\pi^{t-1}} v_{\text{exp}}^{t-1} = \cdots = D_{\pi^t} D_{\pi^{t-1}} \cdots D_{\pi^0} v_{\text{exp}}^{0}.$$

Recall that the matrix D_{π^t} can be written as $\left(\begin{smallmatrix} A_{\pi^t} & B_{\pi^t} \\ \mathbf{0} & \mathbf{1} \end{smallmatrix}\right)$. Since π^t is a proper SD policy with finite values for all t, the submatrix A_{π^t} of D_{π^t} has a spectral radius less than one. There are a finite number of proper SD policies, and thus the maximum of the spectral radii of their corresponding A-matrices is also less than one, denoted by $\hat{\rho}$. Therefore according to (Seneta, 1981; Hartfiel, 2002), we have

$$\lim_{t \to \infty} A_{\pi^t} A_{\pi^{t-1}} \cdots A_{\pi^0} = \mathbf{0}$$

and there exists a nonnegative matrix $\hat{A}$ such that

$$A_{\pi^{t+j}} A_{\pi^{t+j-1}} \cdots A_{\pi^j} \leq \hat{\rho}^{t+1} \hat{A},$$

where we actually choose elements of $\hat{A}$ large enough so that $\mathbf{1} \leq \hat{A}$. There also exists a nonnegative matrix $\hat{B}$ such that for all proper SD policy with finite values π, it holds that $B_\pi \leq \hat{B}$. Now we can show that

$$D_{\pi^t} D_{\pi^{t-1}} \cdots D_{\pi^0} \leq \begin{pmatrix} \hat{\rho}^{t+1} \hat{A} & \frac{1-\hat{\rho}^{t+1}}{1-\hat{\rho}} \hat{A}\hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}.$$

Since it holds for nonnegative matrices that if $A_1 \leq B_1$ and $A_2 \leq B_2$ then $A_1 A_2 \leq B_1 B_2$, we have

$$D_{\pi^0} = \begin{pmatrix} A_{\pi^0} & B_{\pi^0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \leq \begin{pmatrix} \hat{\rho}\hat{A} & \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \leq \begin{pmatrix} \hat{\rho}\hat{A} & \hat{A}\hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix},$$

$$
\begin{aligned}
D_{\pi^1} D_{\pi^0} &= \begin{pmatrix} A_{\pi^1} & B_{\pi^1} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} A_{\pi^0} & B_{\pi^0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} A_{\pi^1} A_{\pi^0} & A_{\pi^1} B_{\pi^0} + B_{\pi^1} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \\
&\le \begin{pmatrix} \hat{\rho}^2 \hat{A} & \hat{\rho} \hat{A} \hat{B} + \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \le \begin{pmatrix} \hat{\rho}^2 \hat{A} & \hat{\rho} \hat{A} \hat{B} + \hat{A} \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} \hat{\rho}^2 \hat{A} & \frac{1-\hat{\rho}^2}{1-\hat{\rho}} \hat{A} \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix},
\end{aligned}
$$

and thus in general

$$
\begin{aligned}
D_{\pi^t} D_{\pi^{t-1}} \cdots D_{\pi^0} = &\begin{pmatrix} A_{\pi^t} & B_{\pi^t} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} A_{\pi^{t-1}} & B_{\pi^{t-1}} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \cdots \begin{pmatrix} A_{\pi^0} & B_{\pi^0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \\
= &\begin{pmatrix} A_{\pi^t} A_{\pi^{t-1}} \cdots A_{\pi^0} & A_{\pi^t} \cdots A_{\pi^1} B_{\pi^0} + A_{\pi^t} \cdots A_{\pi^2} B_{\pi^1} + \cdots + A_{\pi^t} B_{\pi^{t-1}} + B_{\pi^t} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \\
\le &\begin{pmatrix} \hat{\rho}^{t+1} \hat{A} & \hat{\rho}^t \hat{A} \hat{B} + \hat{\rho}^{t-1} \hat{A} \hat{B} + \cdots + \hat{\rho} \hat{A} \hat{B} + \hat{A} \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} \hat{\rho}^{t+1} \hat{A} & \frac{1-\hat{\rho}^{t+1}}{1-\hat{\rho}} \hat{A} \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}.
\end{aligned}
$$

Therefore

$$
\begin{aligned}
\lim_{t\to\infty} v_{\text{exp}}^{t+1} &= \lim_{t\to\infty} D_{\pi^t} D_{\pi^{t-1}} \cdots D_{\pi^0} v_{\text{exp}}^0 \\
&\le \lim_{t\to\infty} \begin{pmatrix} \hat{\rho}^{t+1} \hat{A} & \frac{1-\hat{\rho}^{t+1}}{1-\hat{\rho}} \hat{A} \hat{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} v_{\text{exp}}^0 = \begin{pmatrix} \mathbf{0} & \frac{\hat{A}\hat{B}}{1-\hat{\rho}} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} v_{\text{exp}}^0 = \begin{pmatrix} \mathbf{0} & \frac{\hat{A}\hat{B}}{1-\hat{\rho}} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \bar{v}_{\text{exp}}^0 \\ \mathbf{1} \end{pmatrix} \\
&= \frac{\hat{A} \hat{B} \bar{v}_{\text{exp}}^0}{1 - \hat{\rho}}.
\end{aligned}
$$

That is, for all t, v_{exp}^t is bounded. However, on the other hand, since Condition 3.4 does not hold, there exists a policy $\tilde{\pi}$ such that $v_{\text{exp}}^{\tilde{\pi}}(\tilde{s})$ is unbounded for some state $\tilde{s} \in S \setminus G$. Then it follows that the sequence defined as

$$
\begin{aligned}
\tilde{v}_{\text{exp}}^0(s) &= v_{\text{exp}}^0(s), && s \in S, \\
\tilde{v}_{\text{exp}}^{t+1}(s) &= \sum_{s' \in S} P(s'|s, \tilde{\pi}(s)) \gamma^{r(s,\tilde{\pi}(s),s')} \tilde{v}_{\text{exp}}^t(s'), && s \in S \setminus G,
\end{aligned}
$$

is unbounded for $\tilde{s}$. According to the value update rule, we can show by induction that

$$
v_{\text{exp}}^t(s) \ge \hat{v}_{\text{exp}}^t(s), \qquad s \in S.
$$

The above claim holds for $t = 0$ trivially. Suppose it holds for t. Then

$$v_{\exp}^{t+1}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} v_{\exp}^{t}(s') \geq \sum_{s' \in S} P(s'|s,\tilde{\pi}(s))\gamma^{r(s,\tilde{\pi}(s),s')} v_{\exp}^{t}(s')$$
$$\geq \sum_{s' \in S} P(s'|s,\tilde{\pi}(s))\gamma^{r(s,\tilde{\pi}(s),s')} \tilde{v}_{\exp}^{t}(s') = \tilde{v}_{\exp}^{t+1}(s).$$

In particular, the value $v_{\exp}^{t}(\tilde{s}) \geq \tilde{v}_{\exp}^{t}(\tilde{s})$ is unbounded, which contradicts the earlier claim that $v_{\exp}^{t+1}$ is bounded for all t. Therefore, if Condition 3.4 does not hold, value iteration with Test1 and Test2 will eventually identify an SD policy with infinite values and terminate within a finite time.

We summarize the above discussion in the following theorem.

Theorem 3.5. *Assume that Condition 2.1 and Condition 2.7 hold.*

a. *Policy iteration terminates within a finite time. If Condition 3.4 holds, it terminates with a proper SD-optimal policy; otherwise, it outputs "*`infinite values`*".*

b. *Value iteration with* Test1 *and* Test2 *converges to the optimal values if Condition 3.4 holds; otherwise, it terminates within a finite time and outputs "*`infinite values`*".* □

3.2.3.3 Strictly Negative Models and Condition 3.2

For strictly negative models, if Condition 3.4 does not hold, we need to show that no proper policy has a finite value in order to show that Condition 3.2 (Negative Model with Finite Exponential Utilities) is violated. We do this by keeping track of partial policies that result in infinite values based on Theorem 3.1 and Theorem 3.4.

According to Theorem 3.4, a partial policy results in infinite values if its induced subgraph is a strongly connected with non-zero rewards and its corresponding matrix has a spectral radius great or equal to one. We call such a partial policy a forbidden partial policy. When solving the MDP, we should not consider any policy that is a completion of a forbidden partial policy. We therefore maintain a set of forbidden partial policies and disallow action assignments consistent with these forbidden partial policies.

A minimal forbidden partial policy is a forbidden partial policy where removing any state makes it not forbidden. To maximize the number of policies we can rule out, we search for minimal forbidden partial policies whenever a forbidden partial policy is found. Such minimal policies can be found by enumerating all strongly connected subgraphs of the induced subgraph of the known forbidden partial policy, by starting from those containing only one state and increasing the number of states. Details of the search will be documented elsewhere.

We represent a partial policy as a set of state-action pairs and maintain a set of forbidden policies. Two partial policies can be merged if one is a subset of the other and only the former partial policy is needed. An action is forbidden at a state if this pair of state and action appears in the set of forbidden policies.

In principle, we can go over all partial policies starting from those with only one state. Since a forbidden partial policy can rule out all its completions, we do not need to enumerate all SD policies. However, if Condition 3.2 holds, we still need to go through all policies with finite values, which is intractible. On the other hand, we may favor this approach if we are almost sure no policy has finite values and want to prove it.

Policy iteration only needs to deal with proper policies with infinite values at the initialization step. We start with an arbitrary proper policy and an empty set of forbidden partial policies. If this policy has infinite values (known from solving the policy evaluation equation), we identify its minimal forbidden partial policies and add them to the set. Then we try to obtain a new proper policy that does not contain a known forbidden partial policy. We repeat this procedure until a proper policy with finite values is found or there exists a state for which all actions are forbidden, in which case no policy has finite values. Since there are only a finite number of SD policies, the procedure terminates.

For value iteration, we need to keep track of the set of forbidden policies as we proceed. In each iteration, we obtain the greedy policy and identify its strongly connected components. If any strongly connect component indicates a forbidden partial policy, minimal partial policies are sought and added to the set of forbidden partial policies. When doing value updates, we need to avoid using maximizing actions if they in combination form a forbidden partial policy. This does not affect the convergence of value iteration if Condition 3.2 holds, since on the one hand, a forbidden partial policy eventually drives the values unbounded and cannot be part of a proper SD-optimal policy, and on the other hand, value iteration terminates with any negative initial values. In this way, we also make sure that all forbidden partial policies can be met if no policy has finite values. If we do not avoid forbidden partial policies, the maximizing actions may not change although the finiteness condition is violated. We terminate value iteration if the error between consecutive iterations are close enough (on convergence) or the set of forbidden partial policies indicates that no policy has finite values. This procedure terminates since either we prove no policy has finite values or value iteration converges.

We summarize the above discussion in the following theorem and refer to the procedure of maintaining forbidden partial policies and testing for the negation of Condition 3.2 as Test3.

Theorem 3.6. *Assume that Condition 2.1 and Condition 3.5 hold.*

a. *Policy iteration with* Test3 *terminates within a finite time. If Condition 3.2 holds, it terminates with a proper SD-optimal policy; otherwise, it outputs "*`infinite values`*".*

b. *Value iteration with* Test3 *converges to the optimal values if Condition 3.2 holds; otherwise, it terminates within a finite time and outputs "*`infinite values`*".* □

3.2.3.4 Practical Considerations

In practice, we may not need to test the finiteness conditions at all, or may only test when there are signs that the finiteness conditions are violated. We next consider what can be done for the individual cases we discussed earlier.

First consider positive models. If policy iteration is used, the testing is part of policy iteration (solving the evaluation equations) and no extra work is needed. If value iteration is used, we may test the condition only when there are signs of not converging, for example, when the errors between consecutive iterations keep increasing instead of decreasing.

Now consider strictly negative models. If policy iteration is used and we know there are many proper policies with finite values, we may simply restart policy iteration with a different proper policy, hoping that it has finite values. We can still switch to using the testing after many trials fail. If value iteration is used and we are confident that there is a proper policy with finite values, we can simply perform value iteration without checking for forbidden partial policies since it should converge according to (Patek, 2001). Again, we can still switch to using the testing just to prove that the finiteness condition is violated.

Therefore, to simplify the presentatio in the later parts of this chapter, I will not include testing for finiteness conditions. We can alway incoporate these tests to make the algorithms complete.

3.3 The Representation Transformation

In this section, we discuss in detail the representation transformation approach from (Koenig and Simmons, 1994a,b; Koenig, 1997) in order to fully understand its applicability, advantages, and disadvantages. As illustrated in Figure 3.2, a risk-sensitive planning problem with exponential utility functions is transformed into a risk-neutral planning problem with pseudo-probabilities, which are nonnegative numbers appearing in the places of real probabilities and they do not necessarily sum to one. Then the transformed problem can be solved using risk-neutral planners without any change as long as the planners do not check whether these numbers satisfy the constraints of real probabilities.

In this section, we assume that the rewards can be arbitrary real numbers and the values for all stationary policies exist and are finite.

3.3.1 Definition

For our convenience, we define the representation transformation slightly different from that in (Koenig and Simmons, 1994a,b; Koenig, 1997). Our definition can deal with transformed models both with and without a probabilistic interpretation, thus is a generalization of the original definition. The transformation transforms probabilities and rewards as follows:

$$\bar{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')}, \qquad s,s' \in S, \quad a \in A_s, \tag{3.7}$$

$$\bar{r}(s,a,s') = \begin{cases} 0, & s' \in S \setminus G, \\ \iota, & s' \in G, \end{cases} \qquad s,s' \in S, \quad a \in A_s, \tag{3.8}$$

where $\bar{P}(s'|s,a)$ are known as pseudo-probabilities. In addition, we define the values under an SD policy π in the transformed model with the help of finite horizon values $\bar{v}^{\pi}_{T}(s)$. Since $\bar{P}(s'|s,a)$ are not probabilities, we need to define $\bar{v}^{\pi}_{T}(s)$ starting from scratch. The finite horizon values are defined as

$$\bar{v}^{\pi}_{0}(s) = 0, \qquad\qquad s \in S, \tag{3.9}$$

and for all $T \in \mathbb{N}$,

$$\begin{aligned}
\bar{v}^{\pi}_{T}(s) &= 0, && s \in G, && (3.10)\\
\bar{v}^{\pi}_{T+1}(s) &= \sum_{s' \in S} \bar{P}\big(s'\big|s,\pi(s)\big)\Big(\bar{r}\big(s,\pi(s),s'\big) + \bar{v}^{\pi}_{T}(s')\Big) && &&\\
&= \sum_{s' \in S \setminus G} \bar{P}\big(s'\big|s,\pi(s)\big)\bar{v}^{\pi}_{T}(s') + \iota \sum_{s' \in G} \bar{P}\big(s'\big|s,\pi(s)\big), && s \in S \setminus G, && (3.11)
\end{aligned}$$

where the last step follows from Eq. (3.8) and Eq. (3.10). Then the infinite horizon values are

$$\bar{v}^{\pi}(s) = \lim_{T \to \infty} \bar{v}^{\pi}_{T}(s), \qquad\qquad s \in S. \tag{3.12}$$

Let T approach ∞. Then we obtain from Eq. (3.11) that the values $\bar{v}^{\pi}(s)$ satisfy the following system of policy evaluation equations

$$\begin{aligned}
\bar{v}^{\pi}(s) &= 0, && s \in G, && (3.13)\\
\bar{v}^{\pi}(s) &= \sum_{s' \in S} \bar{P}\big(s'\big|s,\pi(s)\big)\Big(\bar{r}\big(s,\pi(s),s'\big) + \bar{v}^{\pi}(s')\Big) && &&\\
&= \sum_{s' \in S \setminus G} \bar{P}\big(s'\big|s,\pi(s)\big)\bar{v}^{\pi}(s') + \iota \sum_{s' \in G} \bar{P}\big(s'\big|s,\pi(s)\big), && s \in S \setminus G. && (3.14)
\end{aligned}$$

Moreover, we define the optimal values to be

$$\bar{v}^{*}(s) = \max_{\pi \in \Pi^{\mathrm{SD}}} \bar{v}^{\pi}(s), \qquad\qquad s \in S. \tag{3.15}$$

3.3.1.1 Pseudo-Probabilities with a Probabilistic Interpretation

We next show that under the following Condition 3.6, the transformed model has a probabilistic interpretation. This is a generalization of the cases analyzed in (Koenig and Simmons, 1994a,b; Koenig, 1997), where only strictly negative or strictly positive rewards are considered. This interpretation highlights which risk-neutral solution methods can be used together with the representation transformation to solve $\mathsf{MEU}_{\text{exp}}$ problems.

Condition 3.6 (Substochastic Pseudo-Probabilities)**.** For all states $s, s' \in S$ and all actions $a \in A_s$,

$$\sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} \leq 1. \tag{3.16}$$

That is, for all states $s, s' \in S$ and all SD policies $\pi \in \Pi^{\mathrm{SD}}$, the pseudo-probabilities under π form a substochastic matrix.

A special case is that $\gamma^{r(s,a,s')} \leq 1$ for all transitions, which holds when the agent is risk-averse and the original model is positive, or when the agent is risk-seeking and the original model is negative.

When Condition 3.6 (Substochastic Pseudo-Probabilities) holds, we can consider an auxiliary MDP that reduces to the transformed model. The auxiliary MDP has an extra dummy goal state $s_\circ$. All transitions in the transformed model are included in the auxiliary MDP, and for all states $s \in S$ and all actions $a \in A_s$, we have additional transitions to the dummy goal state. Formally, the auxiliary model is defined as follows:

- The state space is $\hat{S} = S \cup \{s_\circ\}$. The set of goal states is $\hat{G} = G \cup \{s_\circ\}$.

- The action space is A, the set of applicable actions at state $s \in S$ is A_s, and $A_{s_\circ} = \{a_{\text{null}}\}$ since $s_\circ$ is a goal state.

- The transition probabilities for all states $s, s' \in S$ and all actions $a \in A_s$ are defined as

$$\hat{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')},$$
$$\hat{P}(s_\circ|s,a) = 1 - \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')}.$$

- The reward function is defined as: for all states $s \in S$, all actions $a \in A_s$, and all next-time states $s' \in S \cup \{s_\circ\}$,

$$\hat{r}(s,a,s') = \begin{cases} 0, & s' \in (S \setminus G) \cup \{s_\circ\}, \\ \iota, & s' \in G. \end{cases}$$

Because of Condition 3.6 (Substochastic Pseudo-Probabilities), the auxiliary MDP is well-defined.

We denote the values of the auxiliary MDP under the MER objective as $\hat{v}$. We need to show that for all SD policies $\pi \in \Pi^{\mathrm{SD}}$ and all states $s \in S$, $\hat{v}^\pi(s) = \bar{v}^\pi(s)$. First, by comparing the auxiliary MDP and the transformed model, we have immediately that for all states $s, s' \in S$ and all actions $a \in A_s$, it holds that

$$\bar{P}(s'|s,a) = \hat{P}(s'|s,a) \qquad \text{and} \qquad \bar{r}(s,a,s') = \hat{r}(s,a,s').$$

Next we show by induction that the finite horizon values are equal for all horizons. Suppose the SD policy is π. Since $s_\circ$ is a goal state in the auxiliary MDP, we have for all $T \in \mathbb{N}$, $\hat{v}^\pi_T(s_\circ) = 0$. When $T = 0$, for all states $s \in S$, $\hat{v}^\pi_0(s) = \bar{v}^\pi_0(s) = 0$ by definition. Suppose it holds for T that for all states $s \in S$, $\hat{v}^\pi_T(s) = \bar{v}^\pi_T(s)$. We have by definition that for all goal states $s \in G$, $\hat{v}_{T+1}(s) = \bar{v}_{T+1}(s) = 0$. For all non-goal states $s \in S \setminus G$,

$$\begin{aligned}
&\hat{v}^\pi_{T+1}(s) = \sum_{s' \in \hat{S}} \hat{P}\big(s'\big|s,\pi(s)\big)\Big(\hat{r}\big(s,\pi(s),s'\big) + \hat{v}^\pi_T(s')\Big) \\
&= \sum_{s' \in S} \hat{P}\big(s'\big|s,\pi(s)\big)\Big(\hat{r}\big(s,\pi(s),s'\big) + \hat{v}^\pi_T(s')\Big) + \hat{P}\big(s_\circ\big|s,\pi(s)\big)\Big(\hat{r}\big(s,\pi(s),s_\circ\big) + \hat{v}^\pi_T(s_\circ)\Big) \\
&= \sum_{s' \in S} \hat{P}\big(s'\big|s,\pi(s)\big)\Big(\hat{r}\big(s,\pi(s),s'\big) + \hat{v}^\pi_T(s')\Big) && \triangleright\ \hat{r}\big(s,\pi(s),s_\circ\big) = 0 \text{ and } \hat{v}^\pi_T(s_\circ) = 0 \\
&= \sum_{s' \in S} \bar{P}\big(s'\big|s,\pi(s)\big)\Big(\bar{r}\big(s,\pi(s),s'\big) + \bar{v}^\pi_T(s')\Big) = \bar{v}^\pi_{T+1}(s). && \triangleright\ \text{definition}
\end{aligned}$$

Therefore, for all states $s \in S$ and all $T \in \mathbb{N}$, it holds that $\hat{v}^\pi_T(s) = \bar{v}^\pi_T(s)$. By taking limits on both sides, we have that for all states $s \in S$, it holds that $\hat{v}^\pi(s) = \bar{v}^\pi(s)$. Therefore, the pseudo-probabilities have a probabilistic interpretation and the transformed model can be viewed as an MDP with an implicitly represented goal state $s_\circ$.

We notice that

- if the agent is risk-seeking, the transformed rewards $\bar{r}(s,a,s')$ only take values 0 and $\iota = 1$, and the transformed model is positive; and

- if the agent is risk-averse, the transformed rewards $\bar{r}(s,a,s')$ only take values 0 and $\iota = -1$, and the transformed model is negative.

Therefore, the auxiliary MDP can be solved using methods for the MER objective. When the agent is risk-seeking, we can use solution methods for positive models, such as value iteration and policy iteration methods. When the agent is risk-averse, we can use solution methods for negative models, which include value iteration, and if there exists a proper optimal policy, policy iteration. However, it is not guaranteed that for all non-goal states s, the resulting optimal value $\bar{v}^*(s)$ equals the risk-sensitive optimal value $v^*_{\text{exp}}(s)$, which will be discussed next.

3.3.1.2 Conditions for the Representation Transformation to Work

The purpose of the representation transformation is that the transformed problem will be solved using an MER-planner without any change. That is, the MER-planner can find or approximate an SD policy $\bar{\pi}^*$ such that $\bar{v}^{\bar{\pi}^*}(s) = \bar{v}^*(s)$ for all states $s \in S$.

Some conditions must be satisfied for the representation transformation to be applicable. First, we must have $v^*_{\text{exp}}(s) = \bar{v}^*(s)$ for all non-goal states $s \in S \backslash G$. Second, there must exist an $\mathsf{MEU}_{\text{exp}}$-optimal policy π^*_{exp} such that $\pi^*_{\text{exp}}(s) = \bar{\pi}^*(s)$ for all non-goal states $s \in S \setminus G$. Last, we need MER algorithms that do not rely on the "probabilities" to sum to one.

Disregarding the presence of pseudo-probabilities, we have noticed that when the agent is risk-seeking, the transformed model is positive, and when the agent is risk-averse, the transformed model is negative. These observations hold even if Condition 3.6 (Substochastic Pseudo-Probabilities) does not hold. Ideally, the MER algorithms for positive and negative models can be used without any change to solve $\mathsf{MEU}_{\text{exp}}$ problems of the above cases respectively by solving the transformed problems. Such algorithms include value iteration and policy iteration, neither of which check whether the "probabilities" sum to one. It is then reduced to verify whether these methods can find or approximate $\bar{v}^*$, the optimal values of the transformed model, and $\bar{\pi}^*$, an SD optimal policy of the transformed model.

In the following subsections, we discuss in detail under what conditions the representation transformation can be applied, and whether the value iteration and policy iteration methods can be used together with the representation transformation to solve $\mathsf{MEU}_{\text{exp}}$ problems.

3.3.2 Planning Problems with Proper Policies Only

Under a proper policy, the agent reaches a goal state with probability one. We show that if all stationary policies are proper, then the representation transformation can be used.

We first show that under a deterministic proper policy, the transformed model and the original model have the same values for non-goal states under their respective planning objectives. For all deterministic proper policies π, the values for the original model under the $\mathsf{MEU}_{\text{exp}}$ objective satisfy the following system of policy evaluation equations,

$$\begin{aligned} v^{\pi}_{\text{exp}}(s) &= \iota, && s \in G, \\ v^{\pi}_{\text{exp}}(s) &= \sum_{s' \in S} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} v^{\pi}_{\text{exp}}(s'), && s \in S \setminus G. \end{aligned}$$

For a non-goal state $s \in S \setminus G$, we have

$$\begin{aligned} v^{\pi}_{\text{exp}}(s) &= \sum_{s' \in S} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} v^{\pi}_{\text{exp}}(s') \\ &= \sum_{s' \in S \setminus G} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} v^{\pi}_{\text{exp}}(s') + \sum_{s' \in G} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} v^{\pi}_{\text{exp}}(s') \\ &= \sum_{s' \in S \setminus G} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} v^{\pi}_{\text{exp}}(s') + \iota \sum_{s' \in G} P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')} \\ &= \sum_{s' \in S \setminus G} \bar{P}(s'|s,\pi(s)) v^{\pi}_{\text{exp}}(s') + \iota \sum_{s' \in G} \bar{P}(s'|s,\pi(s)), \end{aligned}$$

which is the same system of equations as Eq. (3.14), the policy evaluation equations for non-goal states in the transformed model, except that $\bar{v}^*$ is replaced with v^*_{exp}. Recall that we assumed at the beginning of this section that the values for all stationary policies are finite. We also know that if the values under a deterministic proper policy exist and are finite for all non-goal states, then the above system of equations has a unique solution (Patek, 2001). Therefore, for all non-goal states $s \in S \setminus G$, it holds that $\bar{v}^{\pi}(s) = v^{\pi}_{\text{exp}}(s)$.

Suppose further that all stationary policies are proper. A special case is that the model is acyclic (Koenig, 1997). Then for an $\mathsf{MEU}_{\text{exp}}$-optimal policy π^*_{exp}, which is proper, and for all non-goal states $s \in S \setminus G$, it holds that $v^*_{\text{exp}}(s) = v^{\pi^*_{\text{exp}}}_{\text{exp}}(s) = \bar{v}^{\pi^*_{\text{exp}}}(s) \leq \bar{v}^*(s)$. On the other hand, if there exists some policy $\bar{\pi}^*$ such that there exists a non-goal state $s \in S \setminus G$ and $\bar{v}^*(s) = \bar{v}^{\bar{\pi}^*}(s) > \bar{v}^{\pi^*_{\text{exp}}}(s)$, then $\bar{v}^{\bar{\pi}^*}(s) = v^{\bar{\pi}^*}_{\text{exp}}(s) > v^*_{\text{exp}}(s)$, which is impossible. Therefore, for all non-goal states $s \in S \setminus G$, it holds that $v^*_{\exp}(s) = \bar{v}^*(s)$ and there exists a policy $\bar{\pi}^*$ that is optimal in both senses.

Under the assumption that all stationary policies are proper, the MER versions of value iteration and policy iteration can be used to solve $\mathsf{MEU}_{\exp}$ planning problems, since $\bar{v}^\pi(s) = v^\pi_{\exp}(s)$ for all stationary policies π and thus these methods are reduced to their respective $\mathsf{MEU}_{\exp}$ version.

3.3.3 Planning Problems with Improper Policies

If not all stationary policies are proper, the representation transformation may not work. For the convenience of analysis, we assume that there exists a proper optimal policy for the original model under the $\mathsf{MEU}_{\exp}$ objective.

We first consider the case where the agent is risk-seeking. In this case, the representation transformation can be applied. When the agent is risk-seeking, $\iota = 1$ and the optimal $\mathsf{MEU}_{\exp}$-values $v^*_{\exp}(s)$ for non-goal states are positive since we assumed there is a proper optimal policy. If π is improper at s, the value $\bar{v}^\pi(s) = 0$ since non-zero rewards can only be collected when a goal state is reached. But for a proper policy $\pi^*_{\exp}$ that is $\mathsf{MEU}_{\exp}$-optimal, the previous subsection shows that for all non-goal states $s \in S \setminus G$, $\bar{v}^{\pi^*_{\exp}}(s) = v^{\pi^*_{\exp}}_{\exp}(s) > 0$. Therefore for all non-goal states $s \in S \setminus G$ and all policies π that are improper at s, $\bar{v}^*(s) = v^*_{\exp}(s) > \bar{v}^\pi(s) = 0$. That is, an improper policy cannot be optimal for the transformed model. Since the solution methods for the transformed model can find or approximate $\bar{v}^*(s)$ for all non-goal states $s \in S \setminus G$, they will also solve the original $\mathsf{MEU}_{\exp}$ problem.

Next we consider risk-averse agents. In this case, the representation transformation cannot be applied when there are improper policies. When the agent is risk-averse, $\iota = -1$ and the optimal $\mathsf{MEU}_{\exp}$-values $v^*_{\exp}(s)$ for non-goal states are negative. Again, if π is improper at s, the value $\bar{v}^\pi(s) = 0$. But for a proper policy π, the previous subsection shows that for all non-goal states $s \in S \setminus G$, $\bar{v}^\pi(s) = v^\pi_{\exp}(s) < 0$. Therefore for all non-goal states $s \in S \setminus G$ and all policy π that is improper at s, $\bar{v}^*(s) = \bar{v}^\pi(s) = 0 > v^*_{\exp}(s)$. That is, an optimal policy for the transformed model will be improper. The MER value iteration method, when applied to the transformed model, will find or approximate the values $\bar{v}^*(s)$ for all states. The policy iteration method on the other hand cannot be applied if no proper policy is optimal. Thus the representation transformation cannot be used.

3.3.4 Advantages and Limitations of the Representation Transformation

The representation transformation, when applicable, is used as a preprocessing step together with risk-neutral planners to solve risk-sensitive planning problems with an exponential utility function (see Figure 3.2). Another advantage is that it can also deal with problems with both negative and positive rewards when applicable.

However, the representation transformation has severe limitations. It is not applicable to risk-averse agents if there can be improper policies, which is a common case in planning. More importantly, it is not justified to be used together with solution methods other than straightforward value iteration and policy iteration, since usually the state space or the transition probabilities are represented implicitly. This is inadequate for solving large-scale planning problems that require more complicated solution methods, where it is not clear how to define pseudo-probabilities in a systematic way. In the rest of this chapter, I use a transformation-of-algorithms approach to overcome these limitations.

3.4 Transformation of Algorithms

As illustrated in Figure 3.1, I consider a nominal transformation of algorithms that transforms a planning algorithm for a risk-neutral objective to an algorithm for the $\mathsf{MEU}_{\text{exp}}$ objective. The transformation hints on how the risk-neutral algorithms can be adapted to solve $\mathsf{MEU}_{\text{exp}}$ problems. The transformed algorithms need to be justified, which usually can be done in a way parallel to the justification for the original algorithms.

This transformation has two variants, which are best explained using the optimality equations. We use the value iteration method for negative models with goal states to illustrate how these variants help constructing basic computational procedures for the $\mathsf{MEU}_{\text{exp}}$ objective. Recall that the system of optimality equations for finite models under the $\mathsf{MEU}_{\text{exp}}$

objective is

$$\begin{aligned} v^*_{\exp}(s) &= \iota, && s \in G, \\ v^*_{\exp}(s) &= \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')} v^*_{\exp}(s'), && s \in S \setminus G. \end{aligned}$$

First, we have the following mapping for goal states in both variants:

$$\begin{tikzcd} v(s) = 0, \quad s \in G, \arrow[d, leftrightarrow] \\ v_{\exp}(s) = \iota, \quad s \in G. \end{tikzcd} \tag{3.17}$$

The two variants differ for non-goal states. We discuss them separately.

3.4.1 Pseudo-Probability Transformation

The pseudo-probability variant relates an $\mathsf{MEU}_{\exp}$ planner to an MER planner through pseudo-probabilities. This can be shown as follows by comparing for non-goal states the respective systems of optimality equations, which the optimal values satisfy respectively:

$$\begin{tikzcd} v(s) = \max\limits_{a\in A_s} \sum\limits_{s'\in S} & P(s'|s,a) \arrow[d, leftrightarrow] & {[r(s,a,s')} \arrow[d, leftrightarrow] & + & v(s')\ ], \\ v_{\exp}(s) = \max\limits_{a\in A_s} \sum\limits_{s'\in S} & P(s'|s,a)\gamma^{r(s,a,s')} & {[\quad 0} & + & v_{\exp}(s')]. \end{tikzcd} \tag{3.18}$$

Therefore, the rewards $r(s,a,s')$ in the MER planner correspond to 0 in the $\mathsf{MEU}_{\exp}$ planner, and the probabilities in the MER planner correspond to the pseudo-probabilities $P(s'|s,a)\gamma^{r(s,a,s')}$ in the $\mathsf{MEU}_{\exp}$ planner. We use the name pseudo-probability because it does not hold that $\sum\limits_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')} = 1$, and the sum can be greater or less than one. We will also use $\bar{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')}$ to indicate pseudo-probabilities.

Now we consider how the transformation works for the value iteration method for negative models. The value iteration method for negative models under the MER objective uses the following value update rule for non-goal states:

$$v^0(s) = 0, \qquad v^{t+1}(s) = \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a)[r(s,a,s') + v^t(s')].$$

To use the pseudo-probability transformation, we replace $P(s'|s,a)$ with $P(s'|s,a)\gamma^{r(s,a,s')}$ and $r(s,a,s')$ with 0. We also need to replace the initial value 0 with $\iota = U_{\exp}(0)$. The transformation is like this:

$$\begin{array}{llcccc}
v^0(s) = 0, & v^{t+1}(s) & = \max\limits_{a\in A_s}\sum\limits_{s'\in S} & P(s'|s,a) & [r(s,a,s') + & v^t(s')\ ] \\
\Big\downarrow & & & \Big\downarrow & \Big\downarrow & \\
v^0_{\exp}(s) = \iota, & v^{t+1}_{\exp}(s) & = \max\limits_{a\in A_s}\sum\limits_{s'\in S} & P(s'|s,a)\gamma^{r(s,a,s')} & [\quad 0 & +v^t_{\exp}(s')] \\
 & & = \max\limits_{a\in A_s}\sum\limits_{s'\in S} & \multicolumn{3}{l}{P(s'|s,a)\gamma^{r(s,a,s')}v^t_{\exp}(s').}
\end{array}$$

The transformed value update rule therefore is the same as that in the value iteration method for negative models with goal states under the $\mathsf{MEU}_{\exp}$ objective.

3.4.2 Pseudo-Discount Factor Transformation

The other variant relates an $\mathsf{MEU}_{\exp}$ planner to a variably discounted $\mathsf{MER}_{\boldsymbol{\beta}}$ planner but through pseudo-discount factors. This can be shown as follows:

$$\begin{array}{lllcccc}
v(s) & = \max\limits_{a\in A_s}\sum\limits_{s'\in S} P(s'|s,a)[& r(s,a,s') & + & \beta(s,a,s') & v(s')\ ], \\
 & & \Big\updownarrow & & \Big\updownarrow & \\
v_{\exp}(s) & = \max\limits_{a\in A_s}\sum\limits_{s'\in S} P(s'|s,a)[& 0 & + & \gamma^{r(s,a,s')} & v_{\exp}(s')].
\end{array} \tag{3.19}$$

Again, the rewards $r(s,a,s')$ in the $\mathsf{MER}_{\boldsymbol{\beta}}$ planner correspond to 0 in the $\mathsf{MEU}_{\exp}$ planner, but the state-dependent discount factors $\beta(s,a,s')$ in the MER planner correspond to pseudo-discount factors $\gamma^{r(s,a,s')}$. We use the name pseudo-discount factor because $\gamma^{r(s,a,s')}$ may be greater than one.

We continue to use the value iteration method for negative models with goal states to illustrate how this variant works. The value update rule for negative models under the $\mathsf{MER}_{\boldsymbol{\beta}}$ objective is

$$v^{t+1}(s) = \max_{a\in A_s}\sum_{s'\in S} P(s'|s,a)[r(s,a,s') + \beta(s,a,s')v^t(s')],$$

where $\beta(s,a,s') < 1$. To use the pseudo-discount factor transformation, we replace $\beta(s,a,s')$ with $\gamma^{r(s,a,s')}$, and $r(s,a,s')$ with 0:

$$\begin{array}{rlcccl}
v^{t+1}(s) = \max\limits_{a\in A_s}\sum\limits_{s'\in S}P(s'|s,a)[& r(s,a,s') & + & \beta(s,a,s') & v^t(s') &] \\
 & \downarrow & & \downarrow & & \\
v^{t+1}_{\text{exp}}(s) = \max\limits_{a\in A_s}\sum\limits_{s'\in S}P(s'|s,a)[& 0 & + & \gamma^{r(s,a,s')} & v^t_{\text{exp}}(s') &] \\
= \max\limits_{a\in A_s}\sum\limits_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')}v^t_{\text{exp}}(s'). & & & & &
\end{array}$$

Then we obtain the value update rule for negative models under the $\mathsf{MEU}_{\text{exp}}$ objective but without the initial values, since the $\mathsf{MER}_{\boldsymbol{\beta}}$ version of value iteration converges for any initial values as long as $\beta(s,a,s') < 1$ for all transitions. This is not the case for the $\mathsf{MEU}_{\text{exp}}$ objective. This problem can be easily solved since we can require $v^0(s) = 0$ in addition and then replace 0 with ι. Since the $\mathsf{MER}_{\boldsymbol{\beta}}$ version of value iteration does not deal with goal states separately, we need to have special treatments for goal states, which is simple in this example and has been taken care of by setting the initial values to ι. This example also shows that we need to be careful when using the transformation of algorithms, and it is necessary to formally prove the correctness of methods obtained using the transformation of algorithms.

Notice that the system of optimality equations for the MER_{β} objective is a special case of that for the $\mathsf{MER}_{\boldsymbol{\beta}}$ objective where $\beta(s,a,s') = \beta$. We can therefore also start with the MER_{β} objective, which is used by most of the efficient AI planners for which we seek risk-sensitive generalizations, and obtain a version of the planning algorithm under the $\mathsf{MER}_{\boldsymbol{\beta}}$ objective. But caution is needed to place the discount factors β at the right places. A rule of thumb is to make sure the pseudo-discount factors $\gamma^{r(s,a,s')}$ make sense in the context, for example, to make sure the summation indices are in the proper places.

3.4.3 Short Summary

These two variants are almost the same if the probabilities are given explicitly as to be shown in Section 3.5 for LAO*, an important search-based method. But when the probabilities are given implicitly, one variant can be more suitable than the other. The intuition is that the pseudo-probability transformation is more suitable for implicit probabilities that are temporally extended, resulting from temporal abstraction, while the pseudo-discount factor transformation is more suitable for factored MDPs, resulting from state abstraction. We will discuss them in more detail in Section 3.6 and Section 3.7.

I emphasize that these transformations of algorithms are only nominal, in the sense that they provide hints about how to adapt MER methods to solve problems under the $\mathsf{MEU}_{\mathrm{exp}}$ objective. We often need independent proofs of the correctness of the resulting methods. Usually this can also involve identifying conditions under which the transformed algorithms are valid. Fortunately, the proofs for the original MER algorithms often provide helpful ideas about how to construct new proofs and how to identify such conditions.

The transformation of algorithms is used to transform DT planners for large-scale planning problems, and the resulting methods can be proved to be correct. This approach therefore overcomes the limitation of the previous representation transformation approach. The following sections provide further evidence for the applicability of this approach.

3.5 Explicit Probabilities: The LAO* Example

In this section, we use the LAO* method as an example to show that for algorithms using explicit probabilities, either variant of the transformation can be used. We first analyze the original MER version of LAO* and identify the parts that need to be transformed, then present the transformed method and show its correctness. This will also be the pattern when we discuss other methods for solving large-scale MDPs under the $\mathsf{MEU}_{\mathrm{exp}}$ objective.

The LAO* method (Hansen and Zilberstein, 1998, 1999a, 2001) uses heuristic search to solve problems for which Condition 3.5 (Strictly Negative Model) holds, that is, problems with goal states and strictly negative rewards. It can also be combined with structural dynamic programming methods to solve factored MDPs (Feng and Hansen, 2002; Hansen *et al.*, 2002). In this section, we first consider the extension of LAO* to solve flat MDPs under the $\mathsf{MEU}_{\mathrm{exp}}$ objective. In Section 3.7.3, we consider the extension of LAO* to solve factored MDPs.

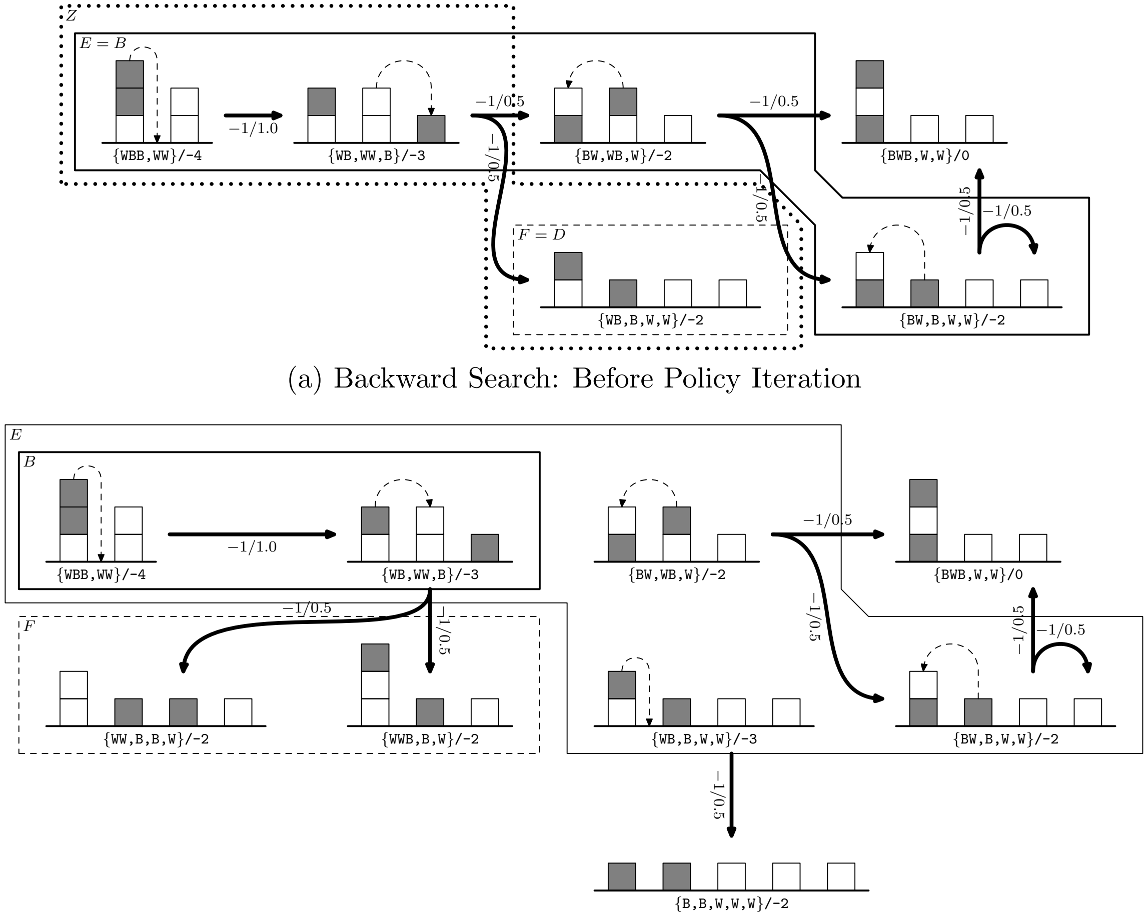
LAO* (Hansen and Zilberstein, 2001) is an offline search method that generalizes the AO* method for searching AND-OR graphs (Nilsson, 1980). MDPs can be viewed as AND-OR graphs, where the states are OR nodes and the actions are AND nodes. For an OR node, the solution depends on exactly one child, which is the max operation in the optimality equations for MDPs. For an AND node, the solution depends on all children, which is the expectation over all possible outcomes in the optimality equations for MDPs.

The benefit of LAO* is that we only need to focus on states that can be reached in the process of seeking the solution, thus avoiding enumerating the complete state space. Another benefit is that we can use heuristic values to speed up planning. Heuristic values are upper bounds of the optimal value functions. For the original risk-neutral LAO*, the heuristic values $h(s)$ are given such that $0 \geq h(s) \geq v^*(s)$.

We use the painted blocks domain to illustrate how LAO* works (see Figure 1.6). Recall that we do not need to distinguish the blocks except for their colors. Hence we can represent the states of a painted blocks problem using the following representation. Use `B` and `W` to represent blocks in black and white, respectively. A tower of blocks is represented by a string of `B` and `W` so that the first letter of the string corresponds to the bottom block and the last letter of the string corresponds to the top block. For example, the tower in the goal configuration can be represented as the string `BWB`. A state, consisting of multiple towers, then is a set of strings. Therefore, the initial configuration in this example can be represented as {`WBB,WW`}. In this representation, the goal configuration corresponds to seven goal states: {`BWB,BB`}, {`BWB,BW`}, {`BWB,WB`}, {`BWB,WW`}, {`BWB,B,B`}, {`BWB,B,W`}, and {`BWB,W,W`}. There are 162 states for a five-block problem in this representation. We will show that fewer states are considered in the solution process using LAO* than using plain dynamic programming methods such as policy iteration or value iteration.

3.5.1 LAO* Search

LAO* performs dynamic programming and heuristic search alternately. The heuristic search is guided by the values and partial policies obtained in dynamic programming, while dynamic programming is restricted to relevant states obtained in the heuristic search.



(a) Backward Search: Before Policy Iteration

(b) Forward Search: After Policy Iteration

Figure 3.5: Backward and forward searches in LAO*

We illustrate the search parts of LAO* in Figure 3.5. We assume that the MDP model of the problem M is represented implicitly. LAO* maintains the explicit graph E that is a submodel of M, and the currently best solution graph B that is a subgraph of E. In our example, we assume that at the beginning of an iteration, $E = B$ and contains four states: {WBB,WW}, {WB,WW,B}, {BW,WB,W}, and {BW,B,W,W}, as shown in Figure 3.5(a). We also show the current best policy π using dashed arrows and the resulting state transitions in thick arrows. It so happens that we do not use any painting action in π. We also label the states with their current values.

Similar to other heuristic search methods like A* or AO*, LAO* maintains F, a set of fringe states that are to be expanded. In fact, we determine F by following the current best policy. Therefore in our example, F contains a single state {WB,B,W,W}, labeled with its heuristic value. The state {BWB,W,W} can also be reached by π, but it is a goal state and will not be expanded in the future. Therefore, we exclude goal states from F.

Next, we choose the single state from F to expand. Since expanding a state can cause the values of its ancestors to change, LAO* performs a backward search to determine a set Z, which contains the state just expanded and all of its ancestor states in the explicit graph E. Only states in Z can change their values. We do not need to include other states since their values cannot be affected by the expansion.

Dynamic programming then is performed only on Z, and consequently, only states in Z can possibly change their current best actions. The details of dynamic programming are not illustrated since there are too many actions and possible resulting states. However, the result of dynamic programming is shown in Figure 3.5(b). We can see that state {WB,B,W,W}, which was a fringe state before dynamic programming and had no action assignment, is assigned the current best action to move the black block on the white block to the table. Moreover, the current best action for state {WB,WW,B} also changed.

After the dynamic programming operation, LAO* performs a forward search to determine the new current best solution graph B. The forward search starts with the initial state {WBB,WW}, and follows the current best policy. As the result of the search, B is changed to contain only states {WBB,WW} and {WB,WW,B}. The search also determines the set of fringe states F, which is the set of states that are not in E, but can be reached from B by following the current best policy.

Algorithm 3.1 (LAOStarPI) shows a version of LAO* that formulates the above procedure and uses policy iteration as the dynamic programming method, while value iteration can also be used with subtle differences in the termination condition for value iteration (Hansen and Zilberstein, 2001). We represent the subgraphs E and B with their respective sets of states. The algorithm starts with no states in the explicit graph E (Line 7), and gradually expands the explicit graph by adding states that can be reached using the currently best policy. The set of fringe states F contains states that are not in E, but can be reached from a state in B within one action following the currently best policy π (with the exception when LAO* starts, where B is empty and F is the set of initial states). In each iteration,

Algorithm 3.1 LAO* with Policy Iteration

π = LAOStarPI(M, S_0, G, h)

Input:

- $M = (S, A, P, r)$, an MDP model;
- S_0, a set of initial states;
- G, a set of goal states, $S_0 \cap G = \varnothing$;
- h, a heuristic function;

Output:

- π, an optimal partial policy, implemented as a set of state-action pairs;

Local:

- v, the value function;
- B, the currently best solution graph;
- E, the explicit subgraph;
- F, the fringe states;
- D, the subset of fringe states to expand;
- Z, the relevant states whose values are updated;

```
 1: for all $s \in G$ do
 2:     $v(s) \leftarrow 0$;
 3: end for
 4: for all $s \in S_0$ do
 5:     $v(s) \leftarrow h(s)$;
 6: end for
 7: $E \leftarrow \varnothing$; $B \leftarrow \varnothing$; $\pi \leftarrow \varnothing$; $F \leftarrow S_0$;
 8: repeat
 9:     select $D \subseteq F$;
10:     for all $s \in \partial D \setminus E$ do
11:         $v(s) \leftarrow h(s)$;
12:     end for
13:     $E \leftarrow E \cup D$;
14:     $Z \leftarrow$ BackwardSearch$(M, D, \pi)$;
15:     $v, \pi \leftarrow$ PolicyIterationRestricted$(M, Z, v)$;
16:     $B, F \leftarrow$ ForwardSearch$(M, S_0, \pi, E)$;
17:     $F \leftarrow F \setminus G$;
18: until $F = \varnothing$;
```

Line 9 first selects D, a subset of the fringe states F. Line 13 expands the explicit graph E with D. We need to initialize in Line 10 the values of states in $\partial D \setminus E$, where ∂D consists of states that are not in D, but can be reached from a state in D by performing one action. Policy iteration (PolicyIterationRestricted, Line 15) is then restricted to Z, the set including only newly expanded states D and their ancestors that are in the best solution graph B. The set Z is identified using BackwardSearch in Line 14. At last, Line 17 uses the updated values to identify the new currently best solution graph in E as well as the new fringe states F through ForwardSearch. The process repeats until F is empty.

The forward and backward search procedures, which are very similar, are included in Algorithm 3.2. These procedures use PolicyImage and PolicyPreImage respectively, where PolicyImage$(source, \pi)$ is the set of states that can be reached from a state in *source* in one action following policy π, and PolicyPreImage$(target, \pi)$ is the set of states from which a state in *target* can be reached in one action following policy π. One way of implementing

Algorithm 3.2 Forward and Backward Search Procedures in LAO*

```
$B, F$ = ForwardSearch($M, S_0, \pi, E$)
$B \leftarrow S_0$;
$F \leftarrow \varnothing$;
$temp \leftarrow B$;
while $temp \neq \varnothing$ do
    $temp \leftarrow$ PolicyImage($temp, \pi$);
    $F \leftarrow F \cup (temp \setminus E)$;
    $temp \leftarrow (temp \cap E) \setminus B$;
    $B \leftarrow B \cup temp$;
end while
```

```
$Z$ = BackwardSearch($M, D, \pi$)
$Z \leftarrow D$;
$temp \leftarrow Z$;
while $temp \neq \varnothing$ do
    $temp \leftarrow$ PolicyPreImage($temp, \pi$) $\setminus Z$;
    $Z \leftarrow Z \cup temp$;
end while
```

```
$target$ = PolicyImage($source$,$\pi$)
$target \leftarrow \varnothing$;
for all $s \in source$ do
    $target \leftarrow target \cup \mathrm{succ}(s, \pi)$;
end for
```

```
$source$ = PolicyPreImage($target$,$\pi$)
$source \leftarrow \varnothing$;
for all $s \in target$ do
    $source \leftarrow source \cup \mathrm{pred}(s, \pi)$;
end for
```

PolicyImage and PolicyPreImage is also shown in Algorithm 3.2. In this implementation, we assume the partial policy π is represented as the set of all possible transitions it specifies. We define $\mathrm{succ}(s, \pi) = \mathrm{succ}(s, \pi(s))$ and $s' \in \mathrm{pred}(s, \pi) \iff s \in \mathrm{succ}(s', \pi)$. We assume that the representation of π allows for efficient calculation of the sets $\mathrm{succ}(s, \pi)$ and $\mathrm{pred}(s, \pi)$.

Table 3.1 shows the complete trace of (the search part of) LAO*. The search starts from the initial state S_0, which only includes {WBB,WW} in the example. We assume the planning objective is MER (see Section 3.5.2). We use a heuristic that is obtained by solving a deterministic relaxation of the problem where the best cases will take place (Bonet, 2002). For example, the heuristic value for state {WBB,WW} is -3, since in the best case, a goal state can be reached with three moving actions. The table shows the values of relevant variables at the end of the main loop (Line 18) of Algorithm 3.1 (LAOStarPI). For simplicity, we only

Table 3.1: Trace of LAO* for the painted-blocks problem

	$D = \{s\}$	$\partial D \setminus E$ (new states only)	Z	E	B with π	F
0				∅	∅	{WBB,WW}
1	{WBB,WW}	{WBBW,W}/-4 {BBB,WW}/-3 {WBB,BW}/-1 {WBB,WB}/-3 {WBB,W,W}/-3 {WBW,WW}/-5 {WWB,WB}/-3 {WWB,WW}/-3 {WB,WW,B}/-2	{WBB,WW}	{WBB,WW}/-3	{WBB,WW}->{WB,WW,B}	{WB,WW,B}

2	{WB,WW,B}	{WBW,B,W}/-2 {WWB,B,W}/-2 {BB,WW,B}/-2 {BB,WW,W}/-3 {BW,WB,B}/-1 {BW,WB,W}/-1 {WB,WB,B}/-3 {WB,WW,W}/-2 {WB,B,W,W}/-2 {WW,WW,B}/-4 {WW,B,B,W}/-2	{WBB,WW}
			{WB,WW,B}
				{WBB,WW}/-3.5 {WB,WW,B}/-2.5	{WBB,WW}->{WB,WW,B}
					{WB,WW,B}->{BW,WB,W}
						{BW,WB,W}
						{WB,B,W,W}

3	{BW,WB,W}	{BWW,WB}/-2 {WBW,BW}/-2 {BB,BW,W}/-1 {BB,WB,W}/-3 {BW,WW,W}/-4 {BW,B,W,W}/-1	{WBB,WW}
			{BW,WB,W}
			{WB,WW,B}
				{WBB,WW}/-3.75 {BW,WB,W}/-1.5 {WB,WW,B}/-2.75	{WBB,WW}->{WB,WW,B}
					{BW,WB,W}->{BWB,W,W}
					{WB,WW,B}->{BW,WB,W}
						{BW,B,W,W}
						{WB,B,W,W}

4	{BW,B,W,W}	{BWW,B,W}/-2 {BB,B,W,W}/-2 {BW,BW,W}/-2 {BW,WW,B}/-1 {BW,B,B,W}/-2 {BW,W,W,W}/-4 {WW,B,W,W}/-4 {B,B,W,W,W}/-2	{WBB,WW}
			{BW,WB,W}
			{BW,B,W,W}
			{WB,WW,B}
				{WBB,WW}/-4 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3	{WBB,WW}->{WB,WW,B}
					{BW,WB,W}->{BWB,W,W}
					{BW,B,W,W}->{BWB,W,W}
					{WB,WW,B}->{BW,WB,W}
						{WB,B,W,W}

5	{WB,B,W,W}	{BB,W,W,W}/-3 {WB,WB,W}/-3 {WB,B,B,W}/-2 {WB,W,W,W}/-5	{WBB,WW}
			{WB,WW,B}
			{WB,B,W,W}
				{WBB,WW}/-4 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3 {WB,B,W,W}/-3	{WBB,WW}->{WB,WW,B}
					{WB,WW,B}->{WWB,B,W}
						{WWB,B,W}
						{WW,B,B,W}

6	{WWB,B,W}	{WWBB,W}/-3 {WWBW,B}/-2 {WBB,B,W}/-2 {WWB,BW}/-1 {WWB,B,B}/-3 {WWB,W,W}/-3 {WWW,B,W}/-5	{WBB,WW}
			{WWB,B,W}
			{WB,WW,B}
				{WBB,WW}/-4 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3 {WB,B,W,W}/-3	{WBB,WW}->{WB,WW,B}
					{WB,WW,B}->{WW,B,B,W}
						{WW,B,B,W}

7	{WW,B,B,W}	{WWW,B,B}/-2 {WW,B,B,B}/-2	{WBB,WW} {WB,WW,B} {WW,B,B,W}	{WBB,WW}/-4 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3.5 {WB,B,W,W}/-3 {WW,B,B,W}/-3	{WBB,WW}->{WBB,BW}	{WBB,BW}
8	{WBB,BW}	{WBBW,B}/-2 {BBB,BW}/-1 {WBB,BB}/-4	{WBB,BW} {WBB,WW}	{WBB,BW}/-1.5 {WBB,WW}/-4 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3.5 {WB,B,W,W}/-3 {WW,B,B,W}/-3	{WBB,WW}->{WBB,W,W}	{WBB,W,W}

9	{WBB,W,W}	{BBB,W,W}/-3 {WBW,W,W}/-5	{WBB,WW}
			{WBB,W,W}
				{WBB,BW}/-1.5 {WBB,WW}/-4.25 {WBB,W,W}/-4 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3.5 {WB,B,W,W}/-3 {WW,B,B,W}/-3	{WBB,WW}->{WWB,WB}
					{WB,WW,B}->{WBW,B,W}
					{WB,B,W,W}->{B,B,W,W,W}
						{WBW,B,W}
						{WWB,WB}
						{B,B,W,W,W}

10	{WBW,B,W}	{WBWB,W}/-3 {WBWW,B}/-3 {BBW,B,W}/-2 {WBW,WB}/-4 {WBW,B,B}/-2	{WBB,WW}
			{WBW,B,W}
			{WB,WW,B}
				{WBB,BW}/-1.5 {WBB,WW}/-4.25 {WBB,W,W}/-4 {WBW,B,W}/-3.5 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3.5 {WB,B,W,W}/-3 {WW,B,B,W}/-3	{WBB,WW}->{WWB,WB}
					{BW,WB,W}->{BWB,W,W}
					{BW,B,W,W}->{BWB,W,W}
					{WB,WW,B}->{BW,WB,W}
					{WB,B,W,W}->{B,B,W,W,W}
						{WWB,WB}
						{B,B,W,W,W}

11	{WWB,WB}	{WWB,BB}/-3 {WWW,WB}/-5	{WBB,WW}
			{WWB,WB}
				{WBB,BW}/-1.5 {WBB,WW}/-4.25 {WBB,W,W}/-4 {WBW,B,W}/-3.5 {WWB,WB}/-3 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-3.5 {WB,B,W,W}/-3 {WW,B,B,W}/-3	{WBB,WW}->{WWB,WB}
					{WWB,WB}->{BWB,WB}
					{BW,WB,W}->{BWB,W,W}
					{BW,B,W,W}->{BWB,W,W}
					{WB,WW,B}->{BW,WB,W}
					{WB,B,W,W}->{B,B,W,W,W}
						{B,B,W,W,W}

12	{B,B,W,W,W}	{B,B,B,W,W}/-2 {B,W,W,W,W}/-5	{WBB,WW}
			{WBB,W,W}
			{WBW,B,W}
			{WB,WW,B}
			{WB,B,W,W}
			{B,B,W,W,W}
				{WBB,BW}/-1.5 {WBB,WW}/-4.5 {WBB,W,W}/-4.5 {WBW,B,W}/-4 {WWB,WB}/-3 {WWB,B,W}/-3 {BW,WB,W}/-2 {BW,B,W,W}/-2 {WB,WW,B}/-4 {WB,B,W,W}/-4 {WW,B,B,W}/-3 {B,B,W,W,W}/-4	{WBB,WW}->{WWB,WB}
					{WWB,WB}->{BWB,WB}
					{BW,WB,W}->{BWB,W,W}
					{BW,B,W,W}->{BWB,W,W}
					{WB,WW,B}->{BW,WB,W}
					{WB,B,W,W}->{BW,WB,W}
						∅

expand one state s ($D = \{s\}$) in each iteration. The column labeled as $\partial D \setminus E$ shows only new states that have not been visited before. The resulting optimal policy is shown in Figure 3.6.

Heuristics play an important role in LAO*. A good heuristic function can speed up planning. For our painted-blocks example, we need only 12 state expansions to obtain an optimal policy using the heuristic obtained from a deterministic relaxation. In contrast, if we use only the zero heuristic function, 52 state expansions are needed. However, in either case, LAO* visits only a small portion of the complete state space, which has 162 states.

Algorithm 3.3 Policy Iteration with a Restricted Domain under the MER Objective

v, π = PolicyIterationRestricted(M, Z, v)

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- $Z \subset S$, the subset of states where the policy will be improved;
- v, the current values, only those values on ∂Z are used;

Output:

- v, the currently best values;
- π, a currently best partial policy;

```
initialize $\pi^0$ to be an arbitrary SD "proper" policy;
$t \leftarrow 0$;
repeat
    solve for $v^t(s)$ from the system of equations
    $v^t(s) = \sum_{s' \in Z \cup \partial Z} P(s'|s, \pi^t(s)) \left[r(s, \pi^t(s), s') + v^t(s')\right], \quad s \in Z;$
    for all $s \in Z$ do
        select $\pi^{t+1}(s) \in \arg\max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s, a) \left[r(s, a, s') + v^t(s')\right]$,
        setting $\pi^{t+1}(s) = \pi^t(s)$ if possible;
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi \leftarrow \pi^t$; $v \leftarrow v^{t-1}$;
```

3.5.2 Risk-Neutral LAO*

The policy iteration procedure with a restricted domain is detailed in Algorithm 3.3 (PolicyIterationRestricted). It is based on the following system of optimality equations:

$$v(s) = \max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s, a)[r(s, a, s') + v(s')], \qquad s \in Z. \tag{3.20}$$

The procedure relies on the values of the boundary states of Z, denoted as ∂Z, which are states that are not in Z but can be reached by performing one action from a state in Z.

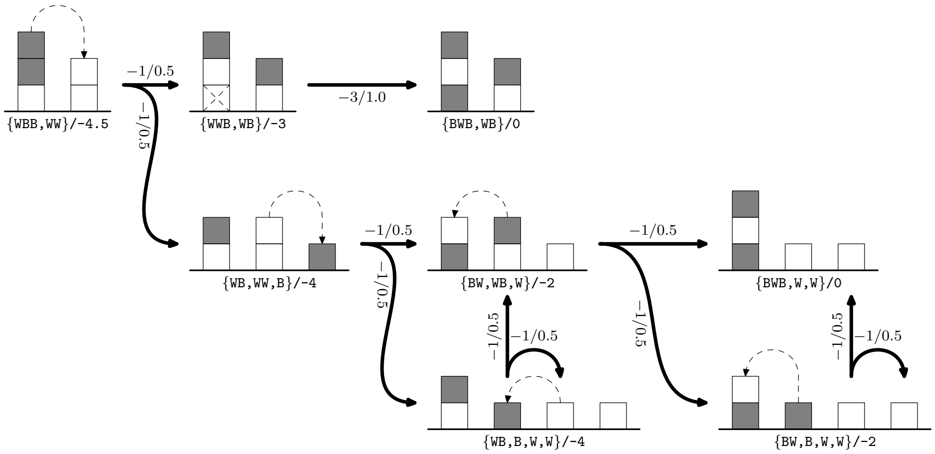


Figure 3.6: An optimal plan obtained by LAO* under the MER objective

We update the values and current best actions of the states in Z. An alternative view is that we can consider states in $S \setminus Z$ as goal states, but the "goal" states in ∂Z have values specified by the current value function v instead of 0. With this view, we require that the initial policy π^0 is proper, in the sense that π^0 will eventually lead to a state outside of Z, which is possible because we assume all rewards are strictly negative.

LAO* uses values in several ways: first, the values of states are initialized based on the heuristic function when these states are added to E; second, the policy iteration method restricted to Z uses the current values of ∂Z to determine the values of states in Z; and third, the forward search for best solution graphs is based on the values indirectly through the currently best known policy. Since the first usage relies on the heuristic function and the third usage does not change the values, only the policy iteration part is relevant to the transformation of algorithms when we extend LAO* to the $\mathsf{MEU}_{\exp}$ objective.

3.5.3 Risk-Sensitive LAO*

Based on the MER version of LAO*, we can apply the pseudo-probability transformation, which replaces $P(s'|s,\pi(s))$ with $P(s'|s,\pi(s))\gamma^{r(s,\pi(s),s')}$ and $r(s,\pi(s),s')$ with 0 in PolicyIterationRestricted. The resulting value iteration procedure is shown in Algorithm 3.4 (PolicyIterationExpRestricted), whose correctness is immediate based on the following system of optimality equations:

$$v_{\exp}(s) = \max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s,a)\gamma^{r(s,a,s')} v_{\exp}(s'), \qquad s \in Z. \tag{3.21}$$

In Algorithm 3.4 (PolicyIterationExpRestricted), the differences from PolicyIterationRestricted are highlighted using boxes. Notice that we need to start with a proper policy with finite values. If we replace PolicyIterationRestricted with PolicyIterationExpRestricted in Algorithm 3.1 (LAOStarPI) (and the heuristic function $h(\cdot)$ with a version suitable for the $\mathsf{MEU}_{\exp}$ objective, to be discussed shortly), we obtain an $\mathsf{MEU}_{\exp}$ version of LAO*, referred to as LAO*Exp.

The pseudo-discount factor transformation can also be used to obtain the same $\mathsf{MEU}_{\exp}$ version of policy iteration on a restricted domain (PolicyIterationExpRestricted) if we start with an MER_{β} version of LAO*, which is described in (Hansen and Zilberstein, 1999b).

Algorithm 3.4 Policy Iteration with a Restricted Domain under the $\mathsf{MEU}_{\exp}$ Objective

$v_{\exp}, \pi = \mathsf{PolicyIterationExpRestricted}(M, Z, v_{\exp}, \gamma)$

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- $Z \subset S$, the subset of states where the policy will be improved;
- $v_{\exp}$, the current values, only those values on ∂Z are used;
- γ, a risk parameter, $\gamma > 0, \gamma \neq 1$;

Output:

- $v_{\exp}$, the currently best values;
- π, a currently best partial policy;

```
$\boxed{\text{initialize } \pi^0 \text{ to be an arbitrary SD proper policy with finite values;}}$
$t \leftarrow 0$;
repeat
    $\boxed{\begin{array}{l}\text{solve for } v^t_{\exp}(s) \text{ from the system of equations} \\ v^t_{\exp}(s) = \sum_{s' \in Z \cup \partial Z} P(s'|s, \pi^t(s)) \gamma^{r(s,\pi^t(s),s')} v^t_{\exp}(s'), \qquad s \in Z;\end{array}}$
    for all $s \in Z$ do
        $\boxed{\begin{array}{l}\text{select } \pi^{t+1}(s) \in \arg\max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s, a) \gamma^{r(s,a,s')} v^t_{\exp}(s'), \\ \text{setting } \pi^{t+1}(s) = \pi^t(s) \text{ if possible;}\end{array}}$
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi \leftarrow \pi^t$; $v_{\exp} \leftarrow v^{t-1}_{\exp}$;
```

Algorithm 3.5 Policy Iteration with a Restricted Domain under the MER_β Objective

$v_\beta, \pi = \mathsf{PolicyIterationDiscountedRestricted}(M, Z, v_\beta, \beta)$

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- $Z \subset S$, the subset of states where the policy will be improved;
- v_β, the current values, only those values on ∂Z are used;
- β, a discount factor, $0 < \beta < 1$;

Output:

- v_β, the currently best values;
- π, a currently best partial policy;

```
initialize $\pi^0$ to be an arbitrary SD policy;
$t \leftarrow 0$;
repeat
    solve for $v^t_\beta(s)$ from the system of equations
    $v^t_\beta(s) = \sum_{s' \in Z \cup \partial Z} P(s'|s, \pi^t(s))[r(s, \pi(s), s') + \beta v^t_\beta(s')], \qquad s \in Z;$
    for all $s \in Z$ do
        select $\pi^{t+1}(s) \in \arg\max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s, a)[r(s, a, s') + \beta v^t_\beta(s')]$,
        setting $\pi^{t+1}(s) = \pi^t(s)$ if possible;
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi \leftarrow \pi^t$; $v_\beta \leftarrow v^{t-1}_\beta$;
```

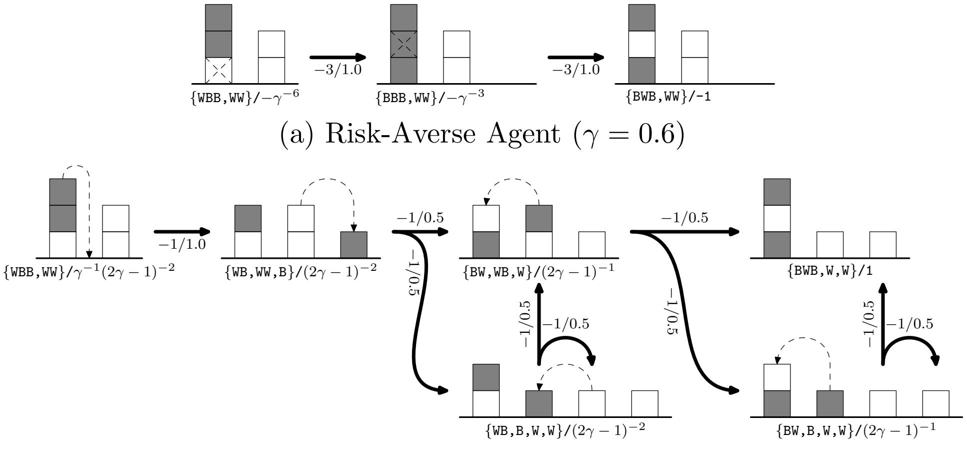
The MER_β version of LAO* differs from the MER version in that it does not require the set of goal states to be non-empty, and uses a version of policy iteration as shown in Algorithm 3.5 ($\mathsf{PolicyIterationDiscountedRestricted}$) that can start with an arbitrary SD policy. It is easy to see that if replacing $r(s, a, s')$ with 0 and β with $\gamma^{r(s,a,s')}$, we basically obtain Algorithm 3.4 ($\mathsf{PolicyIterationExpRestricted}$). But we need to require the initial policy to have more properties (Line 1 in $\mathsf{PolicyIterationExpRestricted}$). Therefore, the two variants of the transformation are equivalent for algorithms using explicit probabilities.

The heuristic function is also important for LAO*Exp. For an $\mathsf{MEU}_{\text{exp}}$ version, the heuristic function should be chosen such that $\iota = U_{\text{exp}}(0) \geq h_{\text{exp}}(s) \geq v^*_{\text{exp}}(s)$. Since in many applications the heuristic function for the MER objective $h(s)$ is derived from a deterministic problem, as we did in Section 3.5.2, we can often use $h_{\text{exp}}(s) = U_{\text{exp}}(h(s)) = \iota\gamma^{h(s)}$.

The correctness and finiteness of LAO*Exp can be proved in the same way as LAO* (Hansen and Zilberstein, 2001). First, we notice that at any point of time, it holds that $v_{\text{exp}}(s) \geq v^*_{\text{exp}}(s)$ for all $s \in S$. This can be shown by induction. Initially, for any state s in the explicit graph E, $v_{\text{exp}}(s) = h_{\text{exp}}(s) \geq v^*_{\text{exp}}(s)$. At any time, suppose it holds for all $s \in E$ that $v_{\text{exp}}(s) \geq v^*_{\text{exp}}(s)$. In the next iteration, the value of a state $s \in E$ is either unchanged (if $s \notin Z$), or its new value is

$$\begin{aligned} v_{\text{exp}}(s) &= \max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s,a)\gamma^{r(s,a,s')} v_{\text{exp}}(s') \\ &\geq \max_{a \in A_s} \sum_{s' \in Z \cup \partial Z} P(s'|s,a)\gamma^{r(s,a,s')} v^*_{\text{exp}}(s') = v^*_{\text{exp}}(s), \end{aligned}$$

where the inequality follows from the induction hypothesis, and the last equality follows from the optimality equation. Second, the algorithm will eventually find a solution graph without fringe states since the model is finite. Last, the correctness of policy iteration on a restricted domain follows directly from the results for models with goal states (see Section 3.2.2). If the agent is risk-seeking, there always exists an optimal policy with finite values. If the agent is risk-averse, with TEST3, the online checking for Condition 3.2 we discussed in Section 3.2.3, LAO*Exp terminates within a finite time. We summarize the above discussion in the following theorem.



(a) Risk-Averse Agent ($\gamma = 0.6$)

(b) Risk-Seeking Agent ($\gamma = 3.0$)

Figure 3.7: Optimal risk-sensitive plans for the painted-blocks problem

Theorem 3.7. *Assume that Condition 2.1 and Condition 3.5 hold. LAO*Exp (with* TEST3 *if the agent is risk-averse) terminates within a finite time. If the inital states have finite values, it terminates with a proper partial SD policy that is optimal for the initial states; otherwise, it outputs "*`infinite values`*".* □

Using LAO*Exp, we can obtain optimal risk-sensitive policies for the painted-blocks problem. For a risk-averse agent with $\gamma = 0.6$, an optimal policy uses only painting actions, as shown in Figure 3.7(a), since the actions are deterministic. For a risk-seeking agent with $\gamma = 3$, an optimal policy uses only moving actions, as shown in Figure 3.7(b), since the total reward is the smallest in the best case. In fact, for all $\gamma \in \left(0, \frac{\sqrt{5}-1}{2}\right)$, the policy shown in Figure 3.7(a) is optimal; for all $\gamma \in \left(\frac{\sqrt{5}+3}{2}, \infty\right)$, the policy shown in Figure 3.7(b) is optimal; and for all $\gamma \in \left(\frac{\sqrt{5}-1}{2}, \frac{\sqrt{5}+3}{2}\right)$, the policy shown in Figure 3.6 is optimal.

3.6 Pseudo-Probability Transformation and Temporally Extended Probabilities

Some planning methods, such as those using temporal abstraction, involve temporally extended probabilities, that is, transition probabilities involving multiple sequentially performed actions. Usually only one-step transition probabilities are explicitly given, and the temporally extended probabilities are implicitly given through a reasoning process. In this section, we show that the pseudo-probability transformation is more suitable in this case.

As examples of temporally extended probabilities, we first discuss in Section 3.6.1 a sensor planning method that uses action sequences, and then discuss in Section 3.6.2 planning with simple options that are partial SD policies. Section 3.6.3 then briefly discusses options as a general model for temporally extended actions, and suggests that methods using options can be transformed to take constant risk attitudes into account.

3.6.1 Pseudo-Probabilities from Action Sequences

We first provide a result concerning transition probabilities under fixed action sequences, which plays an important role in the sensor-planning problem to be discussed next.

To simplify the notation, we assume in this section that $A_s = A$ for all states without loss of generality. Further, we define $A^1 = A$, $A^{n+1} = A^n \times A$, and $A_\infty = \bigcup_{n=1}^{\infty} A^n$. Then A_∞ is the countable set of all action sequences of finite length. Note that A_∞ is different from A^∞, which is the uncountable set of action sequences of an infinite length. For all actions $a \in A$ and all action sequences $\alpha \in A_\infty$, the multi-step transition probability $P(s'|s, \alpha a)$ can be defined recursively as

$$P(s'|s, \alpha a) = \sum_{s'' \in S} P(s''|s, \alpha) P(s'|s'', a). \tag{3.22}$$

The probability $P(\cdot|s, \alpha)$ represents the state distribution after performing the action sequence α starting at state s. We also use $|\alpha|$ to indicate the length of the action sequence α. We have the following simple result that follows from the Markovian property of single-step transition probabilities.

Lemma 3.8. *For all states $s, s' \in S$, all actions $a \in A$, and all action sequences $\alpha \in A_\infty$, we have*

$$P(s'|s, a\alpha) = \sum_{s'' \in S} P(s''|s, a) P(s'|s'', \alpha).$$

Proof. By induction on the length of α. If $\alpha = a'$ for some $a' \in A$, the result holds by definition Eq. (3.22)

$$P(s'|s, aa') = \sum_{s'' \in S} P(s''|s, a) P(s'|s'', a').$$

Suppose the result holds for $\alpha = \alpha' a'$. Consider the action sequence $a\alpha$,

$$\begin{aligned}
P(s'|s, a\alpha) &= P(s'|s, a\alpha' a') \\
&= \sum_{s''' \in S} P(s'''|s, a\alpha') P(s'|s''', a') && \triangleright \text{ By definition of } P(s'|s, a\alpha' a')\text{, see Eq. (3.22)} \\
&= \sum_{s''' \in S} \left(\sum_{s'' \in S} P(s''|s, a) P(s'''|s'', \alpha') \right) P(s'|s''', a') \\
&= \sum_{s''' \in S} \sum_{s'' \in S} P(s''|s, a) P(s'''|s'', \alpha') P(s'|s''', a') \\
&= \sum_{s'' \in S} \sum_{s''' \in S} P(s''|s, a) P(s'''|s'', \alpha') P(s'|s''', a') \\
&= \sum_{s'' \in S} P(s''|s, a) \sum_{s''' \in S} P(s'''|s'', \alpha') P(s'|s''', a') \\
&= \sum_{s'' \in S} P(s''|s, a) P(s'|s'', \alpha' a') && \triangleright \text{ By definition of } P(s'|s'', \alpha' a')\text{, see Eq. (3.22)} \\
&= \sum_{s'' \in S} P(s''|s, a) P(s'|s'', \alpha).
\end{aligned}$$

Therefore, the result holds for all α. □

Corollary 3.9. *For all states $s, s' \in S$, and all action sequences $\alpha = \alpha_1 \alpha_2$ where $\alpha_1, \alpha_2 \in A_\infty$, we have*

$$P(s'|s, \alpha) = \sum_{s'' \in S} P(s''|s, \alpha_1) P(s'|s'', \alpha_2).$$

Proof. Use an induction argument on the length of α_1 similar to that above. □

The pseudo-probability transformation then allows us to define multi-step pseudo-probabilities $\bar{P}(s'|s, \alpha a)$ for all actions $a \in A$ and all action sequences $\alpha \in A_\infty$ as follows:

$$\bar{P}(s'|s, a) = P(s'|s, a) \gamma^{r(s,a,s')}, \tag{3.23}$$

$$\bar{P}(s'|s, \alpha a) = \sum_{s'' \in S} \bar{P}(s''|s, \alpha) \bar{P}(s'|s'', a). \tag{3.24}$$

Moreover, the following results can be seen as obtained through the transformation from Lemma 3.8 and Corollary 3.9.

Lemma 3.10. *For all states $s, s' \in S$, all actions $a \in A$, and all action sequences $\alpha \in A_\infty$, we have*

$$\bar{P}(s'|s, a\alpha) = \sum_{s'' \in S} \bar{P}(s''|s, a) \bar{P}(s'|s'', \alpha).$$

Proof. The proof of Lemma 3.8 applies if all appearances of P are replaced with $\bar{P}$. □

Corollary 3.11. *For all states $s, s' \in S$ and all action sequences $\alpha = \alpha_1\alpha_2$ where $\alpha_1, \alpha_2 \in A_\infty$, we have*

$$\bar{P}(s'|s,\alpha) = \sum_{s'' \in S} \bar{P}(s''|s,\alpha_1)\bar{P}(s'|s'',\alpha_2).$$

In other words, we can generalize the pseudo-probability transformation to include multi-step probabilities such that $P(s'|s,\alpha)$ is replaced with $\bar{P}(s'|s,\alpha)$. The following theorem provides an intuitive meaning for the pseudo-probabilities.

Theorem 3.12. *For all states $s, s' \in S$ and all action sequences $\alpha \in A_\infty$, we have*

$$\bar{P}(s'|s,\alpha) = E^{s,\alpha}\left[\gamma^{\sum\limits_{t=0}^{|\alpha|-1} r_t} \middle| s_{|\alpha|} = s' \right] \cdot P(s'|s,\alpha).$$

Proof. By induction. If $\alpha = a$ for some $a \in A$, then by definition

$$\bar{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')} = E^{s,a}\left[\gamma^{r_0} \middle| s_1 = s'\right] \cdot P(s'|s,a).$$

Suppose the result holds for α. Recall that $w_t = \sum\limits_{\tau=0}^{t-1} r_\tau$. Consider the action sequence αa where $a \in A$. We have

$$\begin{aligned}
&E^{s,\alpha a}\left[\gamma^{w_{|\alpha|+1}} \middle| s_{|\alpha|+1} = s'\right] \\
&\quad = E^{s,\alpha a}_{s''}\left[E^{s,\alpha a}\left[\gamma^{w_{|\alpha|+1}} \middle| s_{|\alpha|+1} = s', s_{|\alpha|} = s''\right] \middle| s_{|\alpha|+1} = s'\right] \\
&\quad = E^{s,\alpha a}_{s''}\left[E^{s,\alpha a}\left[\gamma^{w_{|\alpha|}} \cdot \gamma^{r(s'',a,s')} \middle| s_{|\alpha|+1} = s', s_{|\alpha|} = s''\right] \middle| s_{|\alpha|+1} = s'\right] \\
&\quad = E^{s,\alpha a}_{s''}\left[E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} \middle| s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')} \middle| s_{|\alpha|+1} = s'\right] \\
&\quad = \sum_{s'' \in S} P(s_{|\alpha|} = s''|s,\alpha a, s_{|\alpha|+1} = s')E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} \middle| s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')}
\end{aligned}$$

Therefore,

$$\begin{aligned}
&E^{s,\alpha a}\left[\gamma^{w_{|\alpha|+1}} \middle| s_{|\alpha|+1} = s'\right] \cdot P(s'|s,\alpha a) \\
&\quad = E^{s,\alpha a}\left[\gamma^{w_{|\alpha|+1}} \middle| s_{|\alpha|+1} = s'\right] \cdot P(s_{|\alpha|+1} = s'|s,\alpha a) \\
&\quad = \sum_{s'' \in S} P(s_{|\alpha|} = s''|s,\alpha a, s_{|\alpha|+1} = s')E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} \middle| s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')} \cdot P(s_{|\alpha|+1} = s'|s,\alpha a) \\
&\quad = \sum_{s'' \in S} P(s_{|\alpha|} = s'', s_{|\alpha|+1} = s'|s,\alpha a)E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} \middle| s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')} \\
&\quad = \sum_{s'' \in S} P(s_{|\alpha|} = s''|s,\alpha a)P(s_{|\alpha|+1} = s'|s,\alpha a, s_{|\alpha|} = s'')E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} \middle| s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')}
\end{aligned}$$

$$= \sum_{s'' \in S} P(s''|s,\alpha) P(s'|s'',a) E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} |\, s_{|\alpha|} = s''\right] \cdot \gamma^{r(s'',a,s')}$$

$$= \sum_{s'' \in S} E^{s,\alpha}\left[\gamma^{w_{|\alpha|}} |\, s_{|\alpha|} = s''\right] P(s''|s,\alpha) \cdot \gamma^{r(s'',a,s')} P(s'|s'',a)$$

$$= \sum_{s'' \in S} \bar{P}(s''|s,\alpha) \cdot \bar{P}(s'|s'',a) = \bar{P}(s'|s,\alpha a).$$

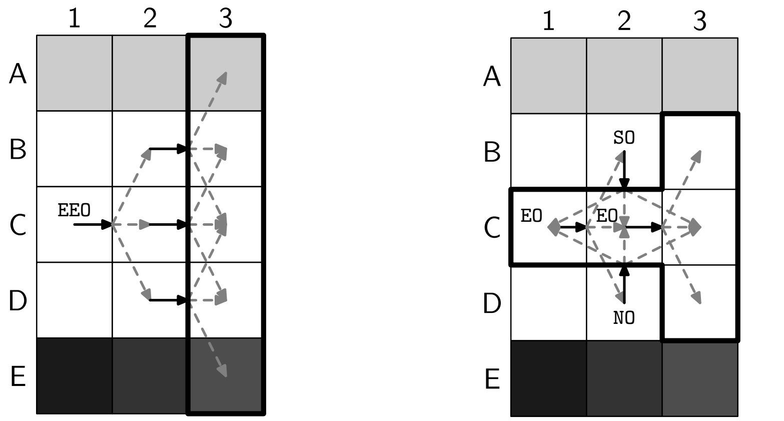
Therefore, the result holds for all action sequences $\alpha \in A_\infty$. □

With the above results, we now consider transforming the sensor-planning method of (Hansen, 1994, 1997) to an $\mathsf{MEU}_{\text{exp}}$ version. The sensor-planning task is modeled as an MDP with observation actions.

3.6.1.1 MDPs with Observation Actions

In this section, we consider a sensor-planning problem, which was originally solved for risk-neutral planning objectives (Hansen, 1994, 1997). We show that the idea from Hansen's method can be used to solve the problem under the $\mathsf{MEU}_{\text{exp}}$ objective (Koenig and Liu, 1999). In this section, we emphasize that the problem can be solved using a nominal transformation so that the original planning algorithm can be easily reused. Therefore, after formulating the problem below, we first present the original planning algorithm, and then discuss how the problem can be solved under the $\mathsf{MEU}_{\text{exp}}$ objective with a correctness proof.

The sensor-planning problem is to determine when to perform costly observation actions (sensing) to obtain complete information about the current state. In MDP models, we assume that the agent knows its current state all the time. In many applications, the agent may need to perform an observation action to get information about its current state, and these actions can be costly. The sensor-planning problem is modeled as an MDP model with observation actions, which we refer to as an MDPwO (MDP with Observations) model. In an MDPwO model, the agent needs to choose whether to perform observation actions, which can give complete information about its current state after a possible state transition. If no observation actions are performed, the agent has no complete information about its current state, and can only infer its current state probabilistically from its previous completely-known state and the actions taken thereafter. If the problem has goal states, we assume



(a) Less Frequent Sensing (b) More Frequent Sensing

Figure 3.8: Segments of two possible sensing plans

that the agent can only stop acting when it knows that its current state is a goal state. Or equivalently, we can consider that the goal states are connected to a virtual goal state, and the agent needs to perform a "stop" action in a goal state to enter the virtual goal state and stop acting. In non-goal states, the "stop" action results in self-looping.

Since we assumed that only observation actions can provide additional information, we require that the action sets are state independent $A_s = A$. Otherwise the agent has some information about its current state. We also assume that there is a set of observation actions $A_o \subseteq A$. If $A_o = A$, the MDPwO model is reduced to a regular MDP model. Therefore in the following, we assume that $A_o \subset A$. We emphasize that the observation actions may also involve state transitions, but the agent will know its exact state after the transition finishes.

An MDPwO can be reformulated as an MDP with an augmented action space $\boldsymbol{A} = {A'}_\infty \times A_o$, where $A' = A \setminus A_o$. According to Lemma 3.8, the multi-step transition probabilities $P(s'|s, \alpha)$ preserve the Markovian property for all $\alpha \in \boldsymbol{A}$. In other words, instead of single actions, we consider action sequences ending with a single observation action, which gives complete information about the current state. Therefore, we can represent a policy as a mapping from states to such action sequences.

We use the robot navigation problem to explain sensor planning. Suppose the robot needs to perform an observation action `O`, which has a cost of 0.2. Figure 3.8 shows segments of two possible sensing plans. For the left sensing plan, if the robot is in state C1, the plan assigns the action sequence EEO, meaning that the robot moves east twice before it senses its current state. Therefore, the robot can be in any of the five locations in the third column, and thus may enter muddy terrain. In contrast, the sensing plan to the right performs the observation action after each movement action. For such a plan, the robot can be in any of the T-shaped set of five locations outlined in Figure 3.8(b) after two movement actions. Therefore, the robot avoids entering muddy terrain, but at the cost of an extra observation action and the possibility of returning to C1. Intuitively, the more risk-averse the robot is, the more often it should sense. With different risk attitudes, there should be qualitatively different optimal policies.

MDPwOs are a special class of Partially Observable MDPs (POMDPs). In POMDP problems, the agent in general cannot know exactly what state it is in. Instead, the agent can only get observations that depend probabilistically on the current state. Different from MDPwOs, these observations often do not provide complete information about the agent's current state. The agent needs to infer which state it is in based on the observation and action history. It is known that the belief state plays an important role for solving POMDPs. A belief state is a probability distribution over all states in the state space, indicating the probabilities of being in a particular state.

General POMDPs are computationally expensive to solve. But for MDPwOs, there exists an efficient method if all rewards are strictly negative, and if the agent needs to reach a goal state. The method uses a combination of heuristic search and policy iteration to solve an MDPwO under risk-neutral planning objectives (Hansen, 1994, 1997). Different from LAO*, which searches in the state space, this method searches for each state in the action space $\boldsymbol{A}$, which is infinite.

3.6.1.2 Maximize Expected Total Rewards

Since we reformulate an MDPwO as an MDP with an augmented action space, Hansen's method for the MER objective can be viewed as performing policy iteration where the plan improvement step uses heuristic search to find the best action sequence in $\boldsymbol{A}$, based on the current values. However, the original treatment did not make this connection explicit. We use the reformulation to clarify how our transformation approach works.

Hansen (1997) showed that for the MER objective, the optimal values satisfy the following optimality equation

$$v^*(s) = 0, \qquad s \in G, \tag{3.25}$$

$$\begin{aligned} v^*(s) &= \max_{\alpha \in \boldsymbol{A}} E^{s,\alpha}\left[\sum_{t=0}^{|\alpha|-1} r_t + v^*\left(s_{|\alpha|}\right)\right] \\ &= \max_{\alpha \in \boldsymbol{A}} \left\{ E^{s,\alpha}\left[\sum_{t=0}^{|\alpha|-1} r_t\right] + \sum_{s' \in S} P(s'|s,\alpha) v^*(s') \right\}, \qquad s \in S \setminus G. \end{aligned} \tag{3.26}$$

Strictly speaking, the maximum in the optimality equation should be replaced by a supremum, since it is possible that the supremum cannot be achieved by any finite sequence. But for the MER objective when all rewards are strictly negative, it is optimal for the agent to sense within a finite number of steps, otherwise the optimal values will be negative infinity.

Hansen (1994, 1997) solved the above system of optimality equations using policy iteration. However, the policy improvement step for policy iteration needs to find an improving action sequence in $\boldsymbol{A}$, an infinite action space. Hansen (1997) showed that the policy improvement step can be implemented using best-first search. For all states $s \in S$ and all action sequences $\alpha \in \boldsymbol{A} \cup A'_\infty$, define

$$f^s(\alpha) = E^{s,\alpha}\left[\sum_{t=0}^{|\alpha|-1} r_t + v\left(s_{|\alpha|}\right)\right].$$

Here we also consider action sequences in A'_∞ since they will appear in the process of the best-first search process. Suppose the current values are $v(s)$ for all states $s \in S$. We need to find an action sequence $\alpha \in \boldsymbol{A}$ such that $f^s(\alpha) > v(s)$ if possible. A key observation in (Hansen, 1997) is that we can decompose the f-values into two parts: for all states $s \in S$ and all action sequences $\alpha \in \boldsymbol{A} \cup A'_\infty$,

$$f^s(\alpha) = g^s(\alpha) + h^s(\alpha),$$

where

$$g^s(\alpha) = E^{s,\alpha}\left[\sum_{t=0}^{|\alpha|-1} r_t\right],$$

$$h^s(\alpha) = E^{s,\alpha}\left[v\left(s_{|\alpha|}\right)\right] = \sum_{s'\in S} P(s'|s,\alpha)v(s').$$

Moreover, the g-values can be calculated recursively as follows

$$\begin{aligned} g^s(a) &= E^{s,a}[r_0] = \sum_{s'\in S} P(s'|s,a)r(s,a,s'), \\ g^s(\alpha a) &= E^{s,\alpha a}\left[\sum_{t=0}^{|\alpha|} r_t\right] = E^{s,\alpha a}\left[\sum_{t=0}^{|\alpha|-1} r_t + r_{|\alpha|}\right] \\ &= E^{s,\alpha a}\left[\sum_{t=0}^{|\alpha|-1} r_t\right] + E^{s,\alpha a}\left[r_{|\alpha|}\right] = g^s(\alpha) + E^{s,\alpha a}\left[r_{|\alpha|}\right]. \end{aligned}$$

Since

$$\begin{aligned} E^{s,\alpha a}\left[r_{|\alpha|}\right] &= E^{s,\alpha a}_{s''}\left[E^{s,\alpha a}\left[r_{|\alpha|}\middle| s_{|\alpha|} = s''\right]\right] = E^{s,\alpha}_{s''}\left[\sum_{s'\in S} P(s'|s'',a)r(s'',a,s')\right] \\ &= \sum_{s''\in S} P(s''|s,\alpha)\sum_{s'\in S} P(s'|s'',a)r(s'',a,s'), \end{aligned}$$

we have

$$g^s(\alpha a) = g^s(\alpha) + \sum_{s''\in S} P(s''|s,\alpha)\sum_{s'\in S} P(s'|s'',a)r(s'',a,s'). \tag{3.27}$$

Similar to the decomposition of the g- and h-values in heuristic search methods such as A*, $g^s(\alpha)$ is the expected reward received for executing the action sequence α starting in state s, and $h^s(\alpha)$ is an estimate of the expected reward that will be received until stopping in a goal state by performing α and then following the current policy.

The decomposition of the g- and h-values makes it possible to use heuristic search methods such as A*. The algorithm using A* search for plan improvement is shown as Algorithm 3.6 (AStarPolicyImprovement). The search starts with an empty action sequence, denoted as ε. Each node in the search graph corresponds to an action sequence. A node is expanded by adding one more action to the sequence, so the children nodes of the current node are of the same prefix and differ only in the last action. Therefore, the search graph

Algorithm 3.6 Policy Improvement Using Heuristic Search under the MER Objective

```
$\alpha, f$ = AStarPolicyImprovement($M, v, s$)
Input:
  • $M = (S, A, P, r)$, an MDPwO model;      • $v$, a value function;
  • $s$, a state where the policy will be improved;
Output:
  • $\alpha$, an action sequence that maximally improves the value of $s$;      • $f$, the value $f^s(\alpha)$;
Local:
  • $OPEN$, a priority queue;

$g(\varepsilon) \leftarrow 0$; $\alpha \leftarrow \varepsilon$;
for all $s' \in S$ do
    $P(s'|s, \varepsilon) \leftarrow 0$;
end for
$P(s|s, \varepsilon) \leftarrow 1$;
repeat
    for all $a \in A$ do
        for all $s' \in S$ do
            $P(s'|s, \alpha a) \leftarrow \sum_{s'' \in S} P(s''|s, \alpha) P(s'|s'', a)$;
        end for
        $g(\alpha a) \leftarrow g(\alpha) + \sum_{s'' \in S} P(s''|s, \alpha) \sum_{s' \in S} P(s'|s'', a) r(s'', a, s')$;
        $h(\alpha a) \leftarrow \sum_{s' \in S} P(s'|s, \alpha a) v(s')$;
        $f(\alpha a) \leftarrow g(\alpha a) + h(\alpha a)$;
        Insert($OPEN, \alpha a, f(\alpha a)$);
    end for
    $\alpha \leftarrow$ Pop($OPEN$);
until $\alpha$ ends with an observation action;
$f \leftarrow f(\alpha)$;
```

actually has a tree structure. In the process of the search, we need to keep track of the g-values and multi-step transition probabilities, which are calculated based on Eq. (3.22) since $\boldsymbol{A} \cup A'_\infty \subset A_\infty$. This information is stored together with the node. *OPEN* is a priority queue, the Insert operation puts an element into the queue along with its f-value as the key, and the Pop operation removes the element with the largest f-value. Since the search graph is of a tree structure, we do not need to check if the priority queue has duplicate nodes. The search terminates when an observation action is used. In fact, since the priority queue has a decreasing order and all rewards are strictly negative, A* search results in a maximally improved f-value, that is, $f^s(\alpha) = \max_{\alpha' \in \boldsymbol{A}} f^s(\alpha')$, where α is the result from the search. Other choices of the search method are possible, as long as they result in $f^s(\alpha) > v(s)$ if possible (Hansen, 1997).

Algorithm 3.7 Policy Iteration for MDPwO under the MER Objective

```
$\pi = \mathsf{PolicyIterationMDPwO}(M)$
Input:
  • $M = (S, A, P, r)$, an MDPwO model;
Output:
  • $\pi$, an optimal policy;

Start with an arbitrary policy $\pi_0$;
$t \leftarrow 0$;
repeat
    Solve the system of linear equations
    $v(s) = 0, \quad s \in G$
    $v(s) = g^s(\pi^t(s)) + \sum_{s' \in S} P(s'|s, \pi^t(s))v(s'), \quad s \in S \setminus G$
    for all $s \in S \setminus G$ do
        $\alpha, f \leftarrow \mathsf{AStarPolicyImprovement}(M, v, s)$;
        if $f > v(s)$ then
            $\pi^{t+1}(s) \leftarrow \alpha$;
        end if
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi \leftarrow \pi^t$;
```

The heuristic search-based policy improvement step then is incorporated into the policy iteration procedure shown in Algorithm 3.7 (PolicyIterationMDPwO). Notice that the policy evaluation equations use g^s-values to simplify the notation. The correctness of the algorithm was shown in (Hansen, 1997).

3.6.1.3 Maximize Expected Exponential Utility of Total Rewards

We now use the pseudo-probability transformation approach to obtain the $\mathsf{MEU}_{\text{exp}}$ version of the sensor planner. Recall that for this transformation, we need to start with an algorithm for the MER objective. The key components in Hansen's method involving probabilities are the definition and decomposition of the f-values as well as the policy evaluation equations. We list the relationships among f, g, h-values as follows:

$$g^s(a) = \sum_{s' \in S} P(s'|s, a) r(s, a, s')$$

$$g^s(\alpha a) = g^s(\alpha) + \sum_{s'' \in S} P(s''|s, \alpha) \sum_{s' \in S} P(s'|s'', a) r(s'', a, s')$$

$$h^s(\alpha) = \sum_{s' \in S} P(s'|s, \alpha) v(s')$$

$$f^s(\alpha) = g^s(\alpha) + h^s(\alpha).$$

Applying the transformation, we obtain

$$\begin{aligned}
g^s_{\text{exp}}(a) &= 0 \\
g^s_{\text{exp}}(\alpha a) &= g^s_{\text{exp}}(\alpha) = 0 \\
h^s_{\text{exp}}(\alpha) &= \sum_{s' \in S} \bar{P}(s'|s,\alpha) v_{\text{exp}}(s') \\
f^s_{\text{exp}}(\alpha) &= g^s_{\text{exp}}(\alpha) + h^s_{\text{exp}}(\alpha) = \sum_{s' \in S} \bar{P}(s'|s,\alpha) v_{\text{exp}}(s').
\end{aligned}$$

They are actually the right formulas for the search of improving action sequences under the $\mathsf{MEU}_{\text{exp}}$ objective, although without any formal justification. Moreover, the policy evaluation equations for the MER objective are

$$\begin{aligned}
v(s) &= 0, && s \in G \\
v(s) &= g^s(\pi^t(s)) + \sum_{s' \in S} P(s'|s,\pi^t(s)) v(s'), && s \in S \setminus G.
\end{aligned}$$

Applying the transformation, we obtain

$$\begin{aligned}
v_{\text{exp}}(s) &= \iota, && s \in G \\
v_{\text{exp}}(s) &= g^s_{\text{exp}}(\pi^t(s)) + \sum_{s' \in S} \bar{P}(s'|s,\pi^t(s)) v_{\text{exp}}(s') \\
&= \sum_{s' \in S} \bar{P}(s'|s,\pi^t(s)) v_{\text{exp}}(s'), && s \in S \setminus G.
\end{aligned}$$

The search and policy iteration algorithms are transformed into Algorithm 3.8 ($\mathsf{AStarPolicyImprovementExp}$) and Algorithm 3.9 ($\mathsf{PolicyIterationExpMDPwO}$), respectively, where the differences are highlighted in boxes.

However, the transformation itself does not prove the correctness of the resulting algorithms. We now provide such a proof. For the $\mathsf{MEU}_{\text{exp}}$ objective, the optimal values need to satisfy the following optimality equation

$$v^*_{\text{exp}}(s) = \iota, \qquad s \in G \tag{3.28}$$

Algorithm 3.8 Policy Improvement Using Heuristic Search under the $\mathsf{MEU}_{\text{exp}}$ Objective

α, f = $\mathsf{AStarPolicyImprovementExp}(M, \gamma, v_{\text{exp}}, s)$

Input:

- $M = (S, A, P, r)$, an MDPwO model;
- γ, a risk parameter, $\gamma > 0, \gamma \neq 1$;
- v_{exp}, a value function;
- s, a state where the policy will be improved;

Output:

- α, an action sequence that maximally improves the value of s;
- f, the value $f^s_{\text{exp}}(\alpha)$;

Local:

- *OPEN*, a priority queue;

```
$f_{\text{exp}}(\varepsilon) \leftarrow v_{\text{exp}}(s)$; $\alpha \leftarrow \varepsilon$;
for all $s' \in S$ do
    $\bar{P}(s'|s, \varepsilon) \leftarrow 0$;
end for
$\bar{P}(s|s, \varepsilon) \leftarrow 1$;
repeat
    for all $a \in A$ do
        for all $s' \in S$ do
            $\boxed{\bar{P}(s'|s, \alpha a) \leftarrow \sum_{s'' \in S} \bar{P}(s''|s, \alpha) P(s'|s'', a) \gamma^{r(s'', a, s')};}$
        end for
        $\boxed{f_{\text{exp}}(\alpha a) \leftarrow \sum_{s' \in S} \bar{P}(s'|s, \alpha a) v_{\text{exp}}(s');}$
        Insert($OPEN, \alpha a, f_{\text{exp}}(\alpha a)$);
    end for
    $\alpha \leftarrow$ Pop($OPEN$);
until $\alpha$ ends with an observation action;
$f \leftarrow f_{\text{exp}}(\alpha)$;
```

$$\begin{aligned} v^*_{\text{exp}}(s) &= \max_{\alpha \in \boldsymbol{A}} E^{s,\alpha} \left[\gamma^{\sum_{t=0}^{|\alpha|-1} r_t} \cdot v^*_{\text{exp}}\left(s_{|\alpha|}\right) \right] \\ &= \max_{\alpha \in \boldsymbol{A}} E^{s,\alpha} \left[\prod_{t=0}^{|\alpha|-1} \gamma^{r_t} \cdot v^*_{\text{exp}}\left(s_{|\alpha|}\right) \right], \qquad s \in S \setminus G. \end{aligned} \tag{3.29}$$

Similar to the previous section, the policy improvement step of policy iteration needs to find an action sequence in $\boldsymbol{A}$ that improves the current value if possible. Suppose the current values are $v_{\text{exp}}(s)$ for all $s \in S$. We need to find an action sequence $\alpha \in \boldsymbol{A}$ such that $f^s_{\text{exp}}(\alpha) > v_{\text{exp}}(s)$ if possible, where we define for all action sequences $\alpha \in \boldsymbol{A} \cup A'_\infty$,

$$f^s_{\text{exp}}(\alpha) = E^{s,\alpha} \left[\left(\prod_{t=0}^{|\alpha|-1} \gamma^{r_t} \right) \cdot v_{\text{exp}}\left(s_{|\alpha|}\right) \right].$$

The following theorem shows that f_{exp} satisfies the property suggested by the pseudo-probability transformation.

Algorithm 3.9 Policy Iteration for MDPwO under the $\mathsf{MEU}_{\exp}$ Objective

$\pi = \mathsf{PolicyIterationExpMDPwO}(M, \gamma, v_{\exp}, s)$

Input:

- $M = (S, A, P, r)$, an MDP model;
- γ, a risk parameter, $\gamma > 0, \gamma \neq 1$;
- $v_{\exp}$, a value function;
- s, a state for which an action sequence which maximally improves its value is seeking;

Output:

- π, an optimal policy;

```
Start with an arbitrary policy $\pi_0$ whose values are finite;
$t \leftarrow 0$;
repeat
    $\boxed{\begin{array}{ll} \text{Solve the system of linear equations} & \\ v_{\exp}(s) = \iota, & s \in G, \\ v_{\exp}(s) = \sum_{s' \in S} \bar{P}(s'|s, \pi^t(s)) v_{\exp}(s'), & s \in S \setminus G, \end{array}}$
    for all $s \in S \setminus G$ do
        $\boxed{\alpha, f \leftarrow \mathsf{AStarPolicyImprovementExp}(M, \gamma, v_{\exp}, s);}$
        if $f > v_{\exp}(s)$ then
            $\pi^{t+1}(s) \leftarrow \alpha$;
        end if
    end for
    $t \leftarrow t + 1$;
until $\pi^t = \pi^{t-1}$;
$\pi \leftarrow \pi^t$;
```

Theorem 3.13. *For all states $s \in S$ and all action sequences $\alpha \in \boldsymbol{A} \cup A'_\infty$, we have*

$$f^s_{\exp}(\alpha) = \sum_{s' \in S} \bar{P}(s'|s, \alpha) v_{\exp}(s').$$

Proof. By induction on the length of α. If $\alpha = a$ for some $a \in A$,

$$f^s_{\exp}(a) = E^{s,a}[\gamma^{r_0} v_{\exp}(s_1)] = \sum_{s' \in S} P(s'|s, a) \gamma^{r(s,a,s')} v_{\exp}(s') = \sum_{s' \in S} \bar{P}(s'|s, a) v_{\exp}(s').$$

Suppose the result holds for α. Then for all actions $a \in A$,

$$\begin{aligned} f^s_{\exp}(a\alpha) &= E^{s,a\alpha}\left[\prod_{t=0}^{|\alpha|} \gamma^{r_t} v_{\exp}\left(s_{|\alpha|+1}\right)\right] = E^{s,a\alpha}_{s''}\left[E^{s,a\alpha}\left[\prod_{t=0}^{|\alpha|} \gamma^{r_t} v_{\exp}\left(s_{|\alpha|+1}\right) \middle| s_1 = s''\right]\right] \\ &= E^{s,a\alpha}_{s''}\left[\gamma^{r_0} \cdot E^{s'',\alpha}\left[\prod_{t=1}^{|\alpha|} \gamma^{r_t} v_{\exp}\left(s_{|\alpha|+1}\right)\right]\right] \\ &= E^{s,a\alpha}_{s''}\left[\gamma^{r_0} f^{s''}_{\exp}(\alpha)\right] = E^{s,a\alpha}_{s''}\left[\gamma^{r_0} \sum_{s' \in S} \bar{P}(s'|s'', \alpha) v_{\exp}(s')\right] \\ &= \sum_{s'' \in S} P(s''|s, a)\left[\gamma^{r(s,a,s'')} \sum_{s' \in S} \bar{P}(s'|s'', \alpha) v_{\exp}(s')\right] \\ &= \sum_{s'' \in S} \bar{P}(s''|s, a) \sum_{s' \in S} \bar{P}(s'|s'', \alpha) v_{\exp}(s') = \sum_{s'' \in S} \sum_{s' \in S} \bar{P}(s''|s, a) \bar{P}(s'|s'', \alpha) v_{\exp}(s') \end{aligned}$$

$$= \sum_{s' \in S} \sum_{s'' \in S} \bar{P}(s''|s,a)\bar{P}(s'|s'',\alpha)v_{\text{exp}}(s') = \sum_{s' \in S} \left(\sum_{s'' \in S} \bar{P}(s''|s,a)\bar{P}(s'|s'',\alpha) \right) v_{\text{exp}}(s')$$
$$= \sum_{s' \in S} \bar{P}(s'|s,a\alpha)v_{\text{exp}}(s').$$

Therefore, the result holds. □

This theorem makes it possible to use best-first search. However, we cannot split the f_{exp}-values into two parts. Algorithm 3.8 (AStarPolicyImprovementExp) is the transformed algorithm, which does not resemble A* any more, but remains a best-first search. The search behaves very similarly to Algorithm 3.6 (AStarPolicyImprovement). The difference is that we instead keep track of the $\bar{P}$ values and no g- nor h-values are needed. The heuristic search-based policy improvement step then is incorporated into the policy iteration procedure shown in Algorithm 3.9 (PolicyIterationExpMDPwO). The policy iteration procedure is correct since there are goal states and all rewards are strictly negative (see Section 3.2).

From the theory of policy iteration, it is sufficient to have the following result concerning improvements in a single step (Patek, 2001).

Theorem 3.14. *If the current policy π is suboptimal, then there exists at least one state s such that Algorithm 3.8 (AStarPolicyImprovementExp) terminates with an action sequence $\alpha \in \boldsymbol{A}$ such that*

$$f^s_{\text{exp}}(\alpha) > v^\pi_{\text{exp}}(s).$$

Proof. Since π is suboptimal, there exists a non-empty set of states for which there exist action sequences that improve the values. We prove the theorem by contradiction. Suppose that Algorithm 3.8 (AStarPolicyImprovementExp) cannot find any of these improving action sequences. Let the action sequence $\alpha = a_1 \cdots a_k \in \boldsymbol{A}$ (at state s) be the shortest action sequence for all states that cannot be found by Algorithm 3.8 (AStarPolicyImprovementExp), but

$$f^s_{\text{exp}}(\alpha) > v^\pi_{\text{exp}}(s).$$

First, we show that if Algorithm 3.8 (AStarPolicyImprovementExp) terminates for s (with an action sequence $\alpha^* \neq \alpha$), then there is a proper prefix $\alpha^0 = a_1 \cdots a_i$ $(1 \leq i < k)$ of α that is in *OPEN*, that is, α^0 has not been expanded. Notice that a_1 was put in *OPEN* at the beginning of the algorithm, and there must be a descendant of a_1 that is in *OPEN* since a_1 has an infinite number of descendants. If there is no proper prefix of α in such action sequences, it must be the case that all proper prefixes of α have been expanded, specifically, $a_1 \cdots a_{k-1}$ has been expanded. But in this case, α itself must have been added to *OPEN* according to the algorithm, and $f^s_{\text{exp}}(\alpha)$ has been calculated, which is greater than $v^\pi_{\text{exp}}(s)$. Then we must have $f^s_{\text{exp}}(\alpha^*) \geq f^s_{\text{exp}}(\alpha) > v^\pi_{\text{exp}}(s)$, and α^* is an improvement. But this contradicts the assumption that an improvement cannot be found by executing Algorithm 3.8 (AStarPolicyImprovementExp) on s. Therefore, we obtain that there is a proper prefix $\alpha^0 = a_1 \cdots a_i$ $(1 \leq i < k)$ of α that is in *OPEN*.

In other words, there exists a proper prefix α^0 such that $\alpha = \alpha^0\alpha'$, and α^0 is in *OPEN*. Since the f^s_{exp} values are monotonically nonincreasing for negative models, the value $f^s_{\text{exp}}(\alpha^0)$ is less than or equal to the value found by Algorithm 3.8 (AStarPolicyImprovementExp), which in turn is less than or equal to $v^\pi_{\text{exp}}(s)$. So, we have

$$f^s_{\text{exp}}(\alpha^0) \leq v^\pi_{\text{exp}}(s) < f^s_{\text{exp}}(\alpha).$$

Expanding both sides, we have

$$\begin{aligned}
\sum_{s' \in S} \bar{P}(s'|s,\alpha^0) v^\pi_{\text{exp}}(s') = f^s_{\text{exp}}(\alpha^0) < f^s_{\text{exp}}(\alpha) &= \sum_{s'' \in S} \bar{P}(s''|s,\alpha) v^\pi_{\text{exp}}(s'') \\
&= \sum_{s'' \in S} \left(\sum_{s' \in S} \bar{P}(s'|s,\alpha^0)\bar{P}(s''|s',\alpha') \right) v^\pi_{\text{exp}}(s'') \\
&= \sum_{s'' \in S} \left(\sum_{s' \in S} \bar{P}(s'|s,\alpha^0)\bar{P}(s''|s',\alpha') v^\pi_{\text{exp}}(s'') \right) \\
&= \sum_{s' \in S} \left(\sum_{s'' \in S} \bar{P}(s'|s,\alpha^0)\bar{P}(s''|s',\alpha') v^\pi_{\text{exp}}(s'') \right) \\
&= \sum_{s' \in S} \bar{P}(s'|s,\alpha^0) \left(\sum_{s'' \in S} \bar{P}(s''|s',\alpha') v^\pi_{\text{exp}}(s'') \right) \\
&= \sum_{s' \in S} \bar{P}(s'|s,\alpha^0) f^{s'}_{\text{exp}}(\alpha').
\end{aligned}$$

Therefore, it must be the case that there exists at least one state s' for which $v^\pi_{\text{exp}}(s') < f^{s'}_{\text{exp}}(\alpha')$, otherwise the above inequality does not hold. But this contradicts that α is the shortest action sequence of this property since α' is a proper subsequence of α. Therefore, the theorem holds. □

We can restore the separation of the g- and h-values if the rewards depend only on the actions, that is, $r(s,a,s') = r(a)$. This is common in planning problems. In this case, we have from Theorem 3.12

$$\bar{P}(s'|s,\alpha) = P(s'|s,\alpha)\gamma^{r(\alpha)}$$

where the reward function of a sequence is defined recursively as $r(\alpha a) = r(\alpha) + r(a)$. Consequently, we have

$$f^s_{\exp}(\alpha) = \sum_{s' \in S} \bar{P}(s'|s,\alpha) v_{\exp}(s') = \gamma^{r(\alpha)} \sum_{s' \in S} P(s'|s,\alpha) v_{\exp}(s').$$

Consider

$$\hat{f}^s(\alpha) = U^{-1}_{\exp}(f^s_{\exp}(\alpha)) = \log_\gamma \left(\iota f^s_{\exp}(\alpha)\right) = r(\alpha) + \log_\gamma \left(\iota \sum_{s' \in S} P(s'|s,\alpha) v_{\exp}(s') \right),$$

which is a monotonic transformation of $f^s_{\exp}(\alpha)$. Consequently, if we use $\hat{f}^s$ instead of $f^s_{\exp}$ in the search, the ordering of elements in the priority queue does not change. Therefore, if we define

$$\begin{aligned} \hat{g}^s(\alpha) &= r(\alpha), \\ \hat{h}^s(\alpha) &= \log_\gamma \left(\iota \sum_{s' \in S} P(s'|s,\alpha) v_{\exp}(s') \right), \end{aligned}$$

we will be able to use A* search again to perform policy improvement. The correctness for this special case is a corollary of Theorem 3.14, but has been proved in a different way in (Koenig and Liu, 1999).

The correctness of the complete algorithm follows from Theorem 3.14 and the theory of policy iteration (Patek, 2001).

Theorem 3.15. *Assume that Condition 2.1 (Finite Model) and Condition 3.5 (Strictly Negative Model) hold. If there exists a sensor policy with finite values, then Algorithm 3.9 (PolicyIterationExpMDPwO) finds an optimal sensor policy within a finite time.* □

Table 3.2 shows optimal sensing policies for two different values of γ. Cells whose action sequences are not used for getting from the start cell (C1) to the goal cell (J1) are left blank. We can see that the risk-averse robot tends to sense more frequently than the risk-seeking robot, as we anticipated. Figure 3.9 shows how often the robots visit each cell during two million runs for three different values of γ, corresponding to the values used in Table 3.2. Here, darker colors indicate a larger number of visits. Two qualitatively different behaviors are observed from these results:

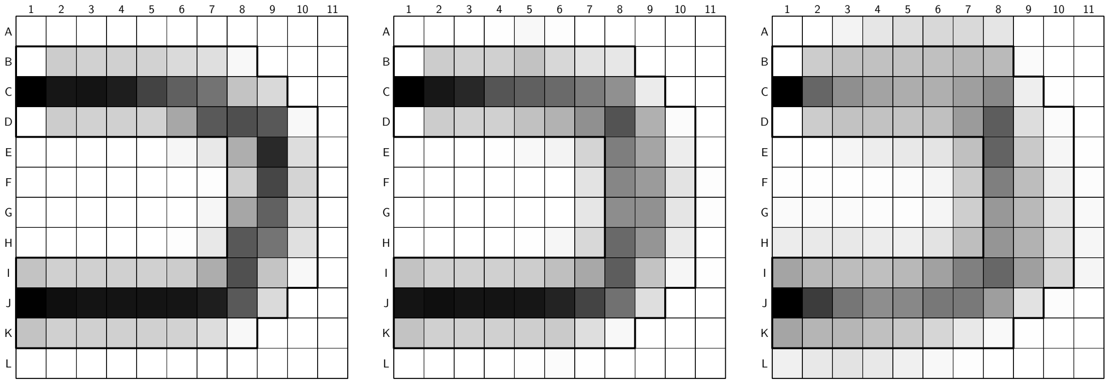
Table 3.2: Optimal risk-sensitive sensing policies

(a) $\gamma = 0.86$ (risk-averse)

	1	2	3	4	5	6	7	8	9	10	11
A								SSSO			
B		SEO	SEO	SEO	SEO	SEO	SO	SSO			
C	EO	EO	EO	EO	EO	EO	SEO	SO	SO	WSO	
D		NEO	NEO	NEO	EEEO	EEO	EO	ESO	SO	WSO	WWSO
E							EESO	ESO	SO	WSO	
F								SSSO	SO	WSO	
G								SSO	SO	WSO	
H						SSWO	SSWO	SO	WSO	WO	
I	SO	SWO	SWO	SWO	SWO	SWO	SWO	SWO	WO	WWO	WWWO
J	goal	WO	WO	WO	WO	WO	WO	WO	NWO		
K	NO	NWO	NWO	NWO	NWO	NWO	NWO	NWO	NWWO		
L						NNWO	NNWO	NNWO			

(b) $\gamma = 1.40$ (risk-seeking)

	1	2	3	4	5	6	7	8	9	10	11
A	SSEEEO			SSEEO	SSEO	SSEO	SSSO	SSSO			
B	SEEEEO			SEEO	SEEO	SEO	SSO	SSO	SSSSSSSWO		
C	EEEEEEESO			EEEESO	EEESO	EESO	SO	SO	SSSSSSWO		
D	EEEEEEESO	EEEEEESO	EEEEESO	EEEESO	EEESO	EESO	ESO	SSSSSWO	SSSSSWO	WSSO	WWSSO
E	NNEEEEO	NEEEEEO	NEEEEO	NEEEO	NEEO	EESSSO	ESSSO	SSSSWO	SSSSWO	WSSO	WWSSO
F	SSSSO	SSSSO	SSSSO	SSSSWO	SSSSWWO	EESSO	ESSSO	SSSWO	SSSWO	WSSO	WWSO
G	SSSO	SSSO	SSSO	SSSWO	SSSWWO	SSSWWWO	SSSWWWWO	SSWO	SSWO	WSO	WWSO
H	SSO	SSO	SSWO	SSWWO	SSWWO	SSWWWO	SSWWWWO	SSWWWWWO	SWO	WWO	WWWO
I	SO	SO	SWO	SWWO	SWWWO	SWWWWO	SWWWWWO	SWWWWWWO	WO	WWO	WWWO
J	goal	WO	WWO	WWWO	WWWWO	WWWWWO	WWWWWWO	WWWWWWWO	WWWWWWWWO	WWWWWWWWWO	WWWO
K	NO	NO	NWO	WWWO	WWWWO	WWWWWO	WWWWWWO	WWWWWWWO	WWWWWWWWO		
L	NNO	NNO	NNWO	NWWO	NWWWO	NWWWWO	NWWWWWO	NWWWWWWO	NWWWWWWWO		



(a) Risk-Averse ($\gamma = 0.86$) (b) Risk-Neutral (c) Risk-Seeking ($\gamma = 1.40$)

Figure 3.9: State-visit frequencies of sensing plans

Table 3.3: Planning complexity

	$\gamma = 0.86$	$\gamma \to 1$	$\gamma = 1.40$
Node expansions	2,071	2,815	5,808
Time (ms) per node expansion	4.6	2.0	5.2

- The figure shows that a more risk-averse robot is more likely to stay on the road and close to the nominal path, which is possible due to the increased sensing frequency. By staying on the road, the robot is likely able to avoid the large costs for getting into the mud, which is the worst-case outcome of an action. However, the worst-case outcome of a policy is to cycle forever and not to be able to achieve the goal cell, thus results in an infinite worst-case cost. Consequently, if a robot is "overly" risk-averse, the expected utility of any plan is negative infinity since it weights the worst-case outcomes more, thus the robot cannot decide how to act in the environment. In our example, this happens when γ is decreased to around 0.80.

- On the other hand, the figure shows that risk-seeking robots do not even attempt to stay in the center of the road all the time. For example, the robot can save two moving actions in the best-case if it stays to the west of the north-south part of the road rather than moving to its center. In general, smaller sensing frequencies and attempts to cut the corners decrease the probability that the robots stay on the road but also decrease the plan-execution cost in the best case. In fact, the action sequences of the start cell get longer and longer as the robots become more risk-seeking until the action sequence is able to move the robots to the goal cell if they are really lucky.

Table 3.3 shows that our sensor planner calculates the optimal sensing policies efficiently, albeit, with some overhead. The number of node expansions depends on the sensing frequency. It increases as the sensing frequency of the optimal sensing policies decreases, because the search trees then need to get searched much deeper (the search depth increases with the number of moving actions between sensing actions). Our sensor planner has a slight run-time disadvantage per node expansion compared to the original sensor planner, because it has to calculate exponentials and logarithms to avoid numerical overflow and underflow, and its heuristic search method cannot calculate the heuristic values quite as efficiently as the original sensor planner.

3.6.1.4 Short Summary

We have shown that the pseudo-probability transformation can be applied to a sensor planning method (Hansen, 1994, 1997) to obtain a risk-sensitive version of the planner. We also showed that the correctness of the resulting planner needs to be proved separately. The optimal sensing plans obtained from the transformed planner are qualitatively different from an risk-neutral optimal plan, which further confirms that risk attitudes can have a significant influence in planning and thus are worth studying. We further showed that the efficiency of the transformed method is comparable to that of the original risk-neutral one.

3.6.2 Pseudo-Probabilities from Simple Options

Another kind of temporally extended probabilities relates to hierarchical methods, or methods using temporal abstraction. These methods were originally developed for reinforcement learning problems (Sutton *et al.*, 1999b), but can be adapted to solve planning problems (Hauskrecht *et al.*, 1998). Instead of finding optimal policies, these methods try to find good policies fast using domain knowledge. We analyze a simple version of such methods in this section using simple options.[4]

A simple option $o = (I_o, \pi_o, T_o)$ is a 3-tuple consisting of a set of initial states I_o, a partial SD policy $\pi_o : S \rightharpoonup A$, and a set of termination states T_o. For the robot navigation example, we can have a simple option that traverses the top horizontal path, as shown in Figure 3.10, where the set I_o is highlighted with green dots and the set T_o with red dashes. The simple option o can be initiated if the agent is in a state $s \in I_o$. The agent then follows π_o until it reaches and terminates in a termination state $s' \in T_o$. Notice that π_o is a partial policy, that is, $\pi_o(s)$ is only defined for a subset of states $S_o \subset S$. Obviously, $I_o \subseteq S_o$. We require that the agent performs at least one action before it stops in a termination state. Therefore, it is possible to have $I_o \cap T_o \neq \varnothing$. We assume that a simple option will

[4] They are called simple to distinguish them from the more general definition of options to be given in Section 3.6.3.

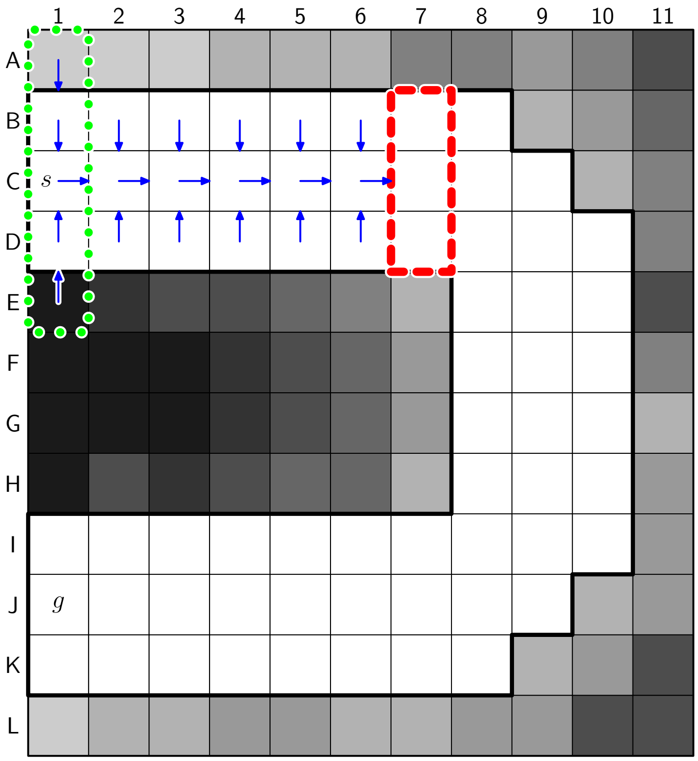


Figure 3.10: A simple option for traversing the top horizontal path

terminate with probability one. For later convenience, we define O to be the set of all simple options, and O_s to be the set of simple options that one can initiate in state s, that is, $O_s = \{o \,|\, s \in I_o, o \in O\}$.

In this section, we show that under the MER and MEU$_{\text{exp}}$ objectives, simple options are similar to regular actions, and thus can be considered as macro actions. Section 3.6.2.1 discusses how planning can be done using simple options. Section 3.6.2.2 discusses one way of obtaining simple options.

3.6.2.1 Hierarchical Planning with Simple Options

It is known that for the MER objective, simple options can be used as macro actions to speed up planning (Precup *et al.*, 1997; Hauskrecht *et al.*, 1998). We will show that simple options can be treated as regular actions, and solution methods such as value iteration and policy iteration can be used to solve problems using simple options.

An action in MDPs is fully specified by the states in which it can be performed, the transition probabilities, and the rewards associated with the state transitions. In fact, a regular action is also a simple option where the set of termination states is the state space S. For this reason, we sometimes refer to the regular actions as primitive actions or primitive options.

A simple option o is a temporally extended action or a macro action a^o, since we can define the three components of an action for option o. The set of states in which the macro action a^o can be performed is I_o. The transition probabilities $P(s'|s, a^o)$ are the temporally extended transition probabilities $P(s'|s, o)$. For the simple option o, its temporally extended transition probabilities are defined as: for all states $s \in S_o$,

$$P(s'|s,o) = \begin{cases} \displaystyle\sum_{t=1}^{\infty} P_t(s'|s,o), & s' \in T_o \\ 0, & \text{otherwise,} \end{cases} \tag{3.30}$$

where $P_t(s'|s, o)$ is the probability that the agent terminates in state $s' \in T_o$ after t epochs under the control of the simple option o while the current state is $s \in S_o$. Notice that we do not require that $s \in I_o$, instead s can be any state that could be entered after the simple option o is initiated. For all simple options $o \in O$, all states $s \in S_o$, and all termination states $s' \in T_o$, these probabilities can be calculated recursively as

$$\begin{aligned} P_1(s'|s,o) &= P(s'|s,\pi_o(s)), \\ P_t(s'|s,o) &= \sum_{s'' \in S \setminus T_o} P(s''|s,\pi_o(s)) P_{t-1}(s'|s'',o), \qquad t \geq 2. \end{aligned}$$

From the definition and the assumption that an option always terminates with probability one, it follows that

$$\sum_{s' \in S} P(s'|s,o) = 1.$$

Therefore, $P(\cdot|s, o)$ is a well-defined probability distribution. For a termination state $s' \in T_o$, the probabilities $P(s'|s, o)$ satisfy the following relationship

$$P(s'|s,o) = P(s'|s,\pi_o(s)) + \sum_{s'' \in S \setminus T_o} P(s''|s,\pi_o(s)) P(s'|s'',o), \qquad s \in S_o.$$

Since the definition of $P(s'|s, o)$ Eq. (3.30) involves an infinite sum, this relationship allows us to determine the transition probabilities for the simple option o by solving the above system of linear equations. For the simple option shown in Figure 3.10, we can obtain the transition probabilities as shown in Table 3.4(a), where the rows indicate states in I_o and the columns indicate states in T_o.

Planning can be done using simple options only if for any simple option $o \in O$, there exists another simple option $o' \in O$ such that $T_o \subseteq I_{o'}$, that is, these two simple options can be "connected" without encountering a state for which no simple option is available. It is often the case that $\bigcup_{o \in O} I_o$ is much smaller than S and thus the state space is reduced, which is one reason that simple options can speed up planning. On the other hand, if the above condition does not hold, we can always use O and A together. Since simple options provide long-term effects, it is the second reason that simple options can speed up planning, as long as their behaviors are similar to "raw" optimal policies over states for which they are defined, and they can be detrimental otherwise (Hauskrecht *et al.*, 1998).

3.6.2.1.1 The Risk-Neutral Case

The reward function for a^o, however, depends on the planning objective. For the MER objective, the reward function for action a^o satisfies

$$r(s, a^o, s') = E^{s,o}\left[\sum_{t=0}^{\tau-1} r_t \middle| s_\tau = s'\right],$$

where $\tau \geq 1$ indicates the random time at which a termination state is entered. This requirement follows from the discussion of a proper reward model in Section 2.3. To determine $r(s, a^o, s')$, we have

$$\begin{aligned}
r(s, a^o, s') &= E^{s,o}\left[\sum_{t=0}^{\tau-1} r_t \middle| s_\tau = s'\right] = E^{s,o}\left[r_0 + \sum_{t=1}^{\tau-1} r_t \middle| s_\tau = s'\right] \\
&= E^{s,o}_{s''}\left[E^{s,o}\left[r_0 + \sum_{t=1}^{\tau-1} r_t \middle| s_1 = s'', s_\tau = s'\right] \middle| s_\tau = s'\right] \\
&= E^{s,o}_{s''}\left[r(s, \pi_o(s), s') + E^{s,o}\left[r_0 + \sum_{t=1}^{\tau-1} r_t \middle| s_1 = s'' \neq s', s_\tau = s'\right] \middle| s_\tau = s'\right] \\
&= r(s, \pi_o(s), s') + E^{s,o}_{s''}\left[E^{s,o}\left[r_0 + \sum_{t=1}^{\tau-1} r_t \middle| s_1 = s'' \neq s', s_\tau = s'\right] \middle| s_\tau = s'\right] \\
&= r(s, \pi_o(s), s') + \sum_{s'' \in S_o \setminus T_o} P(s''|s, o, s_\tau = s')[r(s, \pi_o(s), s'') + r(s'', a^o, s')],
\end{aligned}$$

Table 3.4: Transition probabilities and rewards under the MER objective

	B7	C7	D7
A1	0.184	0.632	0.184
B1	0.184	0.632	0.184
C1	0.184	0.632	0.184
D1	0.184	0.632	0.184
E1	0.184	0.632	0.184

(a) Transition Probabilities

	B7	C7	D7
A1	9.54609	9.82567	9.54609
B1	8.77008	9.04967	8.77008
C1	8.05009	8.32967	8.05009
D1	8.77008	9.04967	8.77008
E1	9.54609	9.82567	9.54609

(b) Equivalent Rewards

and

$$P(s''|s,o,s_{\mathcal{T}}=s') = \frac{P(s'',s_{\mathcal{T}}=s'|s,o)}{P(s_{\mathcal{T}}=s'|s,o)} = \frac{P(s_{\mathcal{T}}=s'|s,o,s_1=s'')P(s_1=s''|s,o)}{P(s'|s,o)}$$

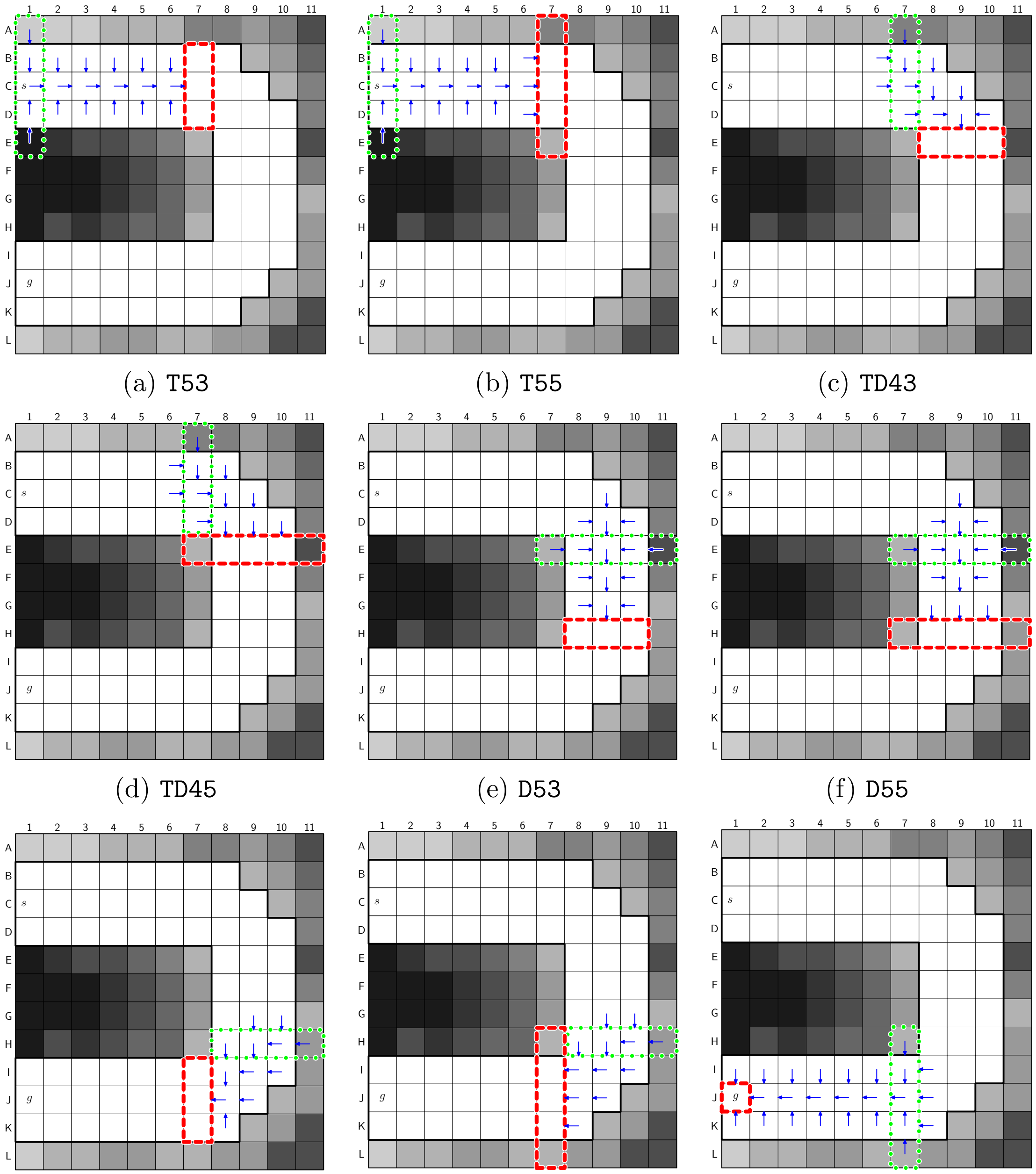
$$= P(s''|s,\pi_o(s))\frac{P(s'|s'',o)}{P(s'|s,o)}.$$

Consequently, the reward function for a^o can be obtained by solving the system of equations for each termination state $s' \in T_o$,

$$r(s,a^o,s') = r(s,\pi_o(s),s') + \sum_{s'' \in S_o \setminus T_o} P(s''|s,\pi_o(s))\frac{P(s'|s'',o)}{P(s'|s,o)}[r(s,\pi_o(s),s'') + r(s'',a^o,s')], \quad s \in S_o.$$

For simplicity, we will refer to $r(s,a^o,s')$ simply as $r(s,o,s')$. For the simple option from Figure 3.10, we can obtain the equivalent rewards as shown in Table 3.4(b).

Precup *et al.* (1997) showed that simple options can be used as regular actions if the transition probabilities $P(s'|s,o)$ and the equivalent reward function $r(s,o,s')$ are known, and value iteration and policy iteration can be used to solve an MDP with simple options. Therefore, MER planners that use simple options first compute the temporally extended transition probabilities $P(s'|s,o)$ and the reward function $r(s,o,s')$, and then use value iteration or policy iteration with simple options to solve the problem. For the robot navigation example, suppose that the available simple options are shown in Figure 3.11. Then we can obtain an "optimal" policy using only options as shown in Figure 3.12, where the expected total reward for the initial state is −29.60, which is slightly lower than that (−28.33) for the optimal policy under the MER objective in Figure 1.10(a). However, value iteration using only options converges after 6 iterations, while it takes 44 iterations to converge using only primitive actions.



(a) T53

(b) T55

(c) TD43

(d) TD45

(e) D53

(f) D55

(g) DB43

(h) DB45

(i) B51

Figure 3.11: Available options

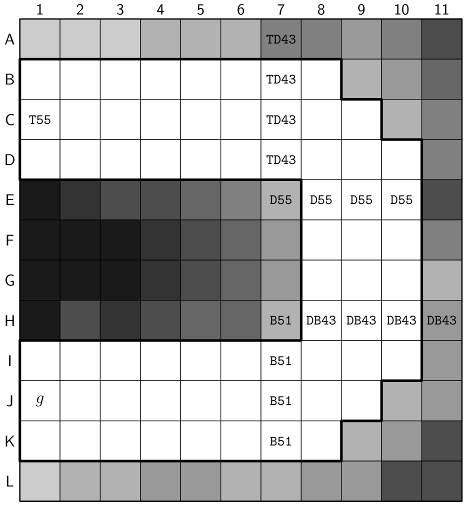


Figure 3.12: An "optimal" policy under the MER objective using simple options only

To obtain $P(s'|s,o)$ and $r(s,o,s')$, we take a direct approach by solving systems of linear equations. This approach was also taken in (Hauskrecht *et al.*, 1998), while in (Precup *et al.*, 1997), it is assumed that $P(s'|s,o)$ and $r(s,o,s')$ are provided along with the simple option o. This process can be viewed as a way of compiling partial policies into actions. Parr (1998) considered a similar task. Our approach differs from his in that his method expresses the values of states in I_o as a linear combination of values of states in T_o, and thus is an implicit representation of the macro action model. We instead prefer an explicit representation of macro actions in terms of transition probabilities and reward functions. With our representation, the pseudo-probability transformation can be applied to obtain a risk-sensitive version of the planning methods.

Higher-level options can be constructed from lower-level options in the same fashion. We thus can have a hierarchy of options that can be used to speed up planning. MAXQ (Dietterich, 2000) and HAM (Parr, 1998), for example, use hierarchies constructed in this way to speed up reinforcement-learning tasks.

3.6.2.1.2 The Risk-Sensitive Case

Now we consider how simple options can be used for the $\mathsf{MEU}_{\text{exp}}$ objective. The main difference is that the reward function needs to be defined differently according to the discussion of the reward model in Section 3.2. We first apply the pseudo-probability transformation and then show the relationship of the results and the macro MDP. Based on the pseudo-probability transformation, we obtain the pseudo-probabilities as: for all states $s \in S_o$,

$$\bar{P}(s'|s,o) = \begin{cases} \sum_{t=1}^{\infty} \bar{P}_t(s'|s,o), & s' \in T_o \\ \\ 0, & \text{otherwise,} \end{cases} \tag{3.31}$$

where for all simple options $o \in O$, all states $s \in S_o$, and all termination states $s' \in T_o$,

$$\begin{aligned} \bar{P}_1(s'|s,o) &= \bar{P}(s'|s,\pi_o(s)) = P(s'|s,\pi_o(s))\gamma^{r(s,\pi_o(s),s')}, \\ \bar{P}_t(s'|s,o) &= \sum_{s'' \in S \setminus T_o} \bar{P}(s''|s,\pi_o(s))\bar{P}_{t-1}(s'|s'',o) \\ &= \sum_{s'' \in S \setminus T_o} P(s''|s,\pi_o(s))\gamma^{r(s,\pi_o(s),s'')}\bar{P}_{t-1}(s'|s'',o), \qquad t \geq 2. \end{aligned}$$

Next, we show that the pseudo-probabilities satisfy the following relationship, which is consistent with our discussion in Section 3.2 of what a proper reward model under the $\mathsf{MEU}_{\text{exp}}$ objective is.

Theorem 3.16. *For all states $s \in S$, all simple options $o \in O$, and all termination states $s' \in T_o$, it holds that*

$$\bar{P}(s'|s,o) = E^{s,o}\left[\gamma^{\sum_{t=0}^{\tau-1} r_t} \middle| s_\tau = s'\right] \cdot P(s'|s,o) = E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s'\right] \cdot P(s'|s,o).$$

According to the above theorem and the discussions in Section 3.2, the equivalent rewards for option o under the $\mathsf{MEU}_{\text{exp}}$ objective can be defined as

$$r(s,o,s') = \log_\gamma \frac{\bar{P}(s'|s,o)}{P(s'|s,o)}. \tag{3.32}$$

Proof. We have that

$$E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s'\right] = \sum_{t=1}^{\infty} E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s', \tau = t\right] P(\tau = t|s,o,s_\tau = s'),$$

and

$$\begin{aligned} E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s'\right] \cdot P(s'|s,o) &= E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s'\right] \cdot P(s_\tau = s'|s,o) \\ &= \sum_{t=1}^{\infty} E^{s,o}\left[\gamma^{w_\tau} \middle| s_\tau = s', \tau = t\right] \cdot P(s_\tau = s', \tau = t|s,o) \end{aligned}$$

$$= \sum_{t=1}^{\infty} E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t\right] \cdot P_t(s'|s,o).$$

Therefore, it is sufficient to show that

$$\bar{P}_t(s'|s,o) = E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t\right] \cdot P_t(s'|s,o).$$

We prove this by induction. When $t = 1$, the result holds by definition. Suppose that the result holds for $t \geq 1$. Then

$$\begin{aligned}
&E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t+1\right] \\
&\quad = \sum_{s'' \in S \setminus T_o} E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t+1, s_1 = s''\right] \cdot P^{s,o}(s_1 = s''|s_\tau = s', \tau = t+1) \\
&\quad = \sum_{s'' \in S \setminus T_o} E^{s,o}\left[\gamma^{r_0}\gamma^{w_{1,\tau}} \,|\, s_\tau = s', \tau = t+1, s_1 = s''\right] \cdot P^{s,o}(s_1 = s''|s_\tau = s', \tau = t+1) \\
&\quad = \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P^{s,o}(s_1 = s''|s_\tau = s', \tau = t+1),
\end{aligned}$$

where $w_{1,\tau} = w_\tau - w_1 = w_\tau - r_0$ and $\tau' = \tau - 1$ is the random termination time of o starting from s''. Since

$$P^{s,o}(s_1 = s''|s_\tau = s', \tau = t+1) = P(s_1 = s''|s_\tau = s', \tau = t+1, s, o),$$

we have

$$\begin{aligned}
&E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t+1\right] \cdot P_{t+1}(s'|s,o) \\
&= E^{s,o}\left[\gamma^{w_\tau} \,|\, s_\tau = s', \tau = t+1\right] \cdot P(s_\tau = s', \tau = t+1|s,o) \\
&= \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P^{s,o}(s_1 = s''|s_\tau = s', \tau = t+1) \\
&\qquad \cdot P(s_\tau = s', \tau = t+1|s,o) \\
&= \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P(s_1 = s'', s_\tau = s', \tau = t+1|s,o) \\
&= \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P(s_1 = s''|s,o) P(s_\tau = s', \tau = t+1|s_1 = s'', o) \\
&= \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} P(s_1 = s''|s,o) E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P(s_\tau = s', \tau = t+1|s_1 = s'', o) \\
&= \sum_{s'' \in S \setminus T_o} \gamma^{r(s,\pi_o(s),s'')} P(s_1 = s''|s,o) \cdot E^{s'',o}\left[\gamma^{w_{\tau'}} \,|\, s_{\tau'} = s', \tau' = t\right] \cdot P(s_{\tau'} = s', \tau' = t|s'', o) \\
&= \sum_{s'' \in S \setminus T_o} \bar{P}(s''|s, \pi_o(s)) \bar{P}_t(s'|s'', o) = \bar{P}_{t+1}(s'|s,o)
\end{aligned}$$

Therefore, the result holds. □

Table 3.5: Pseudo-probabilities and rewards under the $\mathsf{MEU}_{\text{exp}}$ objective

	B7	C7	D7
A1	42.7766	144.4910	42.7766
B1	28.1012	94.9203	28.1012
C1	18.9841	64.1243	18.9841
D1	28.1012	94.9203	28.1012
E1	42.7766	144.4910	42.7766

(a) Pseudo-Probabilities: Risk-Averse

	B7	C7	D7
A1	10.66670	10.63390	10.66670
B1	9.84413	9.81138	9.84413
C1	9.07632	9.04357	9.07632
D1	9.84413	9.81138	9.84413
E1	10.66670	10.63390	10.66670

(b) Equivalent Rewards: Risk-Averse

	B7	C7	D7
A1	0.000443096	0.00154651	0.000443096
B1	0.000716679	0.00250130	0.000716679
C1	0.001119490	0.00390722	0.001119490
D1	0.000716679	0.00250130	0.000716679
E1	0.000443096	0.00154651	0.000443096

(c) Pseudo-Probabilities: Risk-Seeking

	B7	C7	D7
A1	8.69787	8.67476	8.69787
B1	8.00416	7.98110	8.00416
C1	7.36071	7.33764	7.36071
D1	8.00416	7.98110	8.00416
E1	8.69787	8.67476	8.69787

(d) Equivalent Rewards: Risk-Seeking

Moreover, we can calculate the pseudo-probabilities in a way similar to regular probabilities by solving the following system of equations.

Theorem 3.17. *For all simple options $o \in O$, all states $s \in S_o$, and all termination states $s' \in T_o$, it holds that*

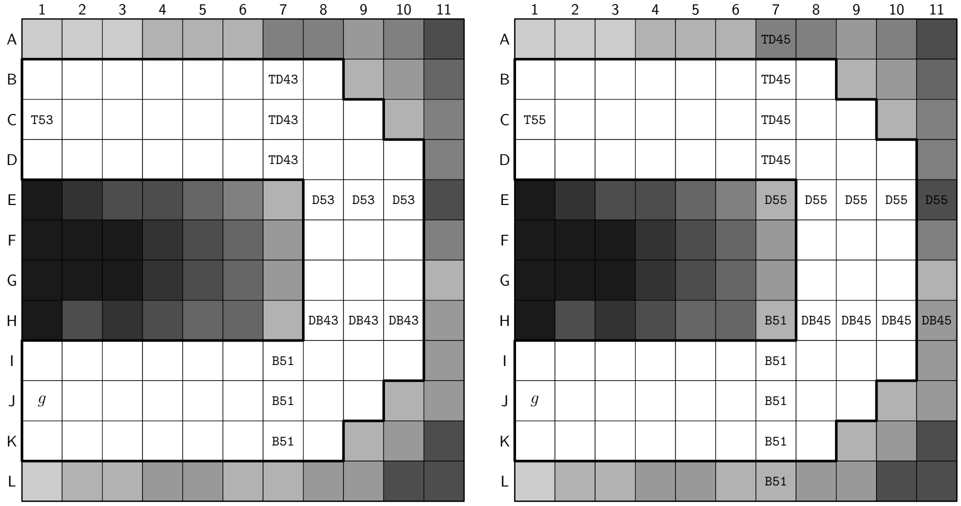
$$\bar{P}(s'|s,o) = \bar{P}(s'|s,\pi_o(s)) + \sum_{s'' \notin T_o} \bar{P}(s''|s,\pi_o(s))\bar{P}(s'|s'',o). \tag{3.33}$$

Proof. We have

$$\begin{aligned}
\sum_{s'' \notin T_o} \bar{P}(s''|s,\pi_o(s))\bar{P}(s'|s'',o) &= \sum_{s'' \notin T_o} \bar{P}(s''|s,\pi_o(s)) \sum_{t=1}^{\infty} \bar{P}_t(s'|s'',o) \\
&= \sum_{t=1}^{\infty} \sum_{s'' \notin T_o} \bar{P}(s''|s,\pi_o(s))\bar{P}_t(s'|s'',o) \\
&= \sum_{t=1}^{\infty} \bar{P}_{t+1}(s'|s,o) = \bar{P}(s'|s,o) - \bar{P}_1(s'|s,o).
\end{aligned}$$

Therefore the result holds. □

Consider the simple option shown in Figure 3.10. If $\gamma = 0.6$ (risk-averse), we can obtain the pseudo-probabilities as shown in Table 3.5(a), which correspond to the equivalent reward function shown in Table 3.5(b) defined by Eq. (3.32). If $\gamma = 2.0$ (risk-seeking), we can obtain the pseudo-probabilities as shown in Table 3.5(b), which correspond to the equivalent reward function shown in Table 3.5(c) defined by Eq. (3.32). Comparing to the risk-neutral case



(a) Risk-Averse (b) Risk-Seeking

Figure 3.13: "Optimal" policies under the $\mathsf{MEU}_{\text{exp}}$ objective using simple options only

Table 3.4(b), we can see that the equivalent rewards are higher for risk-averse agents, and lower for risk-seeking agents.

MER planners using simple options can be transformed into $\mathsf{MEU}_{\text{exp}}$ planners with the pseudo-probability transformation. Notice that for the $\mathsf{MEU}_{\text{exp}}$ planner, we only need to solve one system of equations for $\bar{P}(s'|s,o)$, instead of two systems of equations as in the MER case. If we use the set of options from Figure 3.11, the resulting policies are shown in Figure 3.13(a) and Figure 3.13(b), corresponding to the cases $\gamma = 0.6$ and $\gamma = 2.0$, respectively.

However, there remains a minor issue for using simple options under the $\mathsf{MEU}_{\text{exp}}$ objective. It is possible that the infinite summation in Eq. (3.31) does not converge to a finite value. In this case, the system of equations Eq. (3.33) does not have a finite solution. Since we consider only negative or positive models, infinite pseudo-probabilities can arise if the model is strictly negative and the agent is risk-averse, or if the model is positive and the agent is risk-seeking. This is not a problem, though. In the former case, a policy using the simple option cannot be optimal or ϵ-optimal for any given ϵ, therefore this simple option can be safely ignored. In the latter case, a policy using the simple option would also result in positive infinite values, which however contradicts Condition 3.4 (Positive Model with Finite Exponential Utilities).

We can also use the online testing for policies with infinite values described in Section 3.2.3 to obtain a complete algorithm that obtains a good policy using only simple options if existing and terminates with "`infinite values`" if no such policy has finite values.

3.6.2.2 Determination of Simple Options

In hierarchical planning, the hierarchy of abstract actions (simple options) encodes prior knowledge about the planning problem. Often simple options may not be fully specified. Instead, only the initial set and termination set are given. To make the distinction, we call a pair of the initial set and the termination set a simple pre-option. Such scenarios have been discussed in different contexts (Dean and Lin, 1995; Hauskrecht *et al.*, 1998; Dietterich, 2000).

In this case, we still use $o = (I_o, T_o)$ to indicate a simple pre-option, where the partial policy is not yet defined. We can often determine a region of influence of the option S_o such that the resulting option can only possibly visit states in S_o, while it is also possible that some states in S_o will not be visited, but the states in S_o will help to determine the simple option. In this sense, S_o is slightly different from the same notation used in Section 3.6.2.1.

We first consider the simple case where the hierarchy has only two levels. The high level uses only options, and the low level uses only primitive actions. Under the MER objective, the optimality equations for the high level that uses only options are

$$\begin{aligned} v(s) &= 0, && s \in G, \\ v(s) &= \max_{o \in O_s} \sum_{s' \in S} P(s'|s,o)[r(s,o,s') + v(s')], && s \notin G, \end{aligned}$$

where O_s is the set of all possible options that are consistent with the description of the pre-options. For the low level that uses primitive actions, the optimality equations are

$$\begin{aligned} v^o(s) &= v(s), && s \in T_o \\ v^o(s) &= \max_{\substack{a \in A_s \\ \mathrm{succ}(s,a) \subseteq S_o}} \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^o(s')], && s \in S_o \setminus T_o. \end{aligned}$$

In order to solve the problem and determine the options at the same time, the planner alternates between the low level and the high level until it converges. We refer to this method as hierarchical dynamic programming.

We can apply this method to the robot navigation example. Suppose that we have a set of pre-options that have the same I_o and T_o sets as the simple options shown in Figure 3.11. Then the MER version of hierarchical dynamic programming determines a set of options shown in Figure 3.14 and an "optimal" policy using these options shown in Figure 3.15. The resulting options improve the original definition in Figure 3.11, and the resulting policy achieves an expected reward of -28.36, which is very close to the optimal value -28.33. Notice that it is possible that some of the termination states will not be entered when the algorithm terminates, which indicates that hierarchical dynamic programming improves the initial "guessing" of pre-options needed for solving the problem. For example, state A7 is not entered in option T55′, neither is state E11 in option D55′.

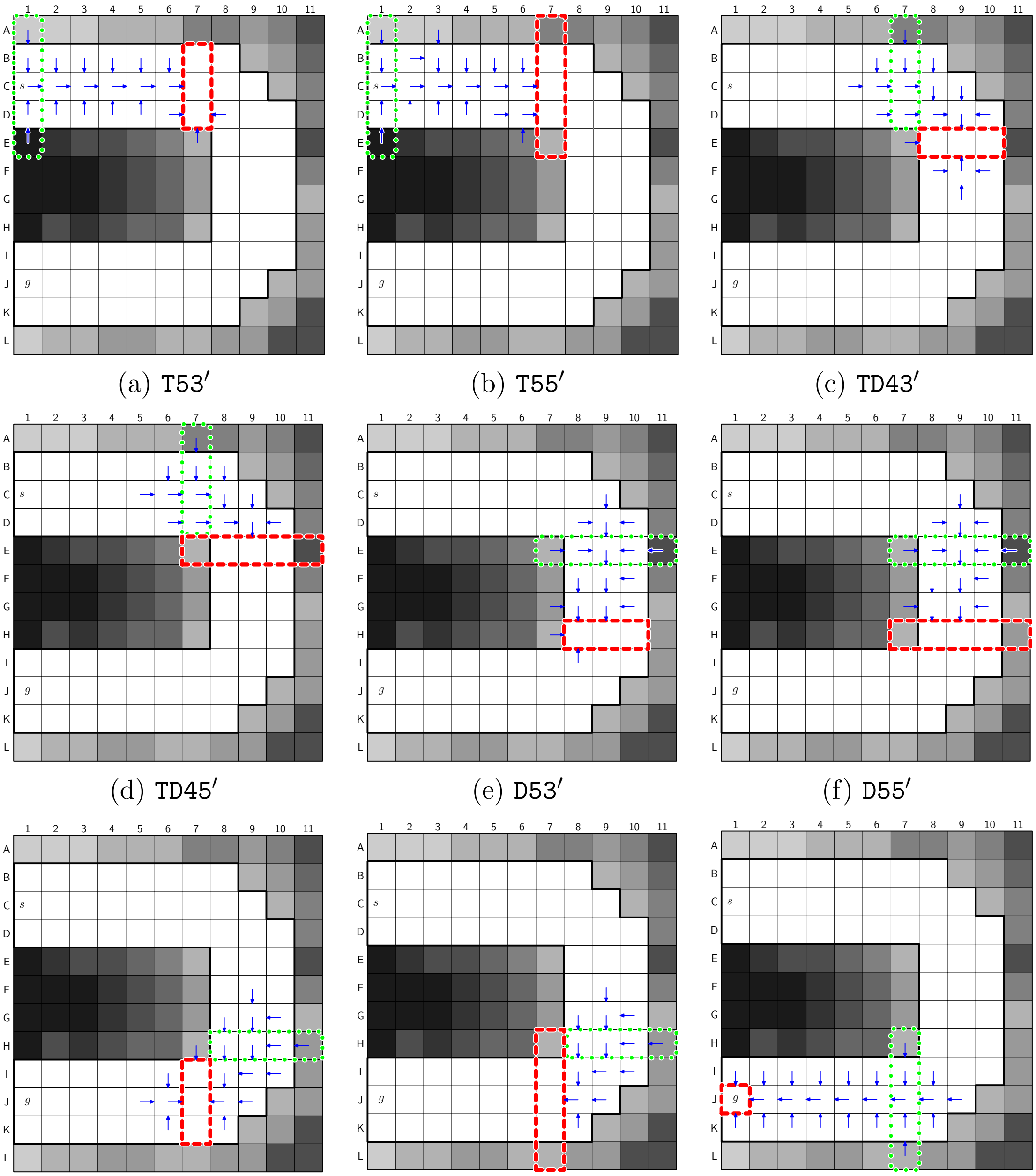
As we have mentioned earlier, options can be organized in a hierarchy, where higher level options use lower level options. A policy that satisfies these systems of optimality equations simultaneously is called recursively optimal (Dietterich, 2000). Although a recursively optimal policy may not be optimal, it is often a good approximation. The above method for two levels of options and primitive actions can be generalized to a hierarchy of options, where the computation starts at the bottom of the hierarchy and proceeds bottom-up.

Under the $\mathsf{MEU}_{\mathrm{exp}}$ objective, we have the following system of optimality equations

$$\begin{aligned}
v_{\mathrm{exp}}(s) &= \iota, && s \in G,\\
v_{\mathrm{exp}}(s) &= \max_{o \in O_s} \sum_{s' \in S} \bar{P}(s'|s,o) v_{\mathrm{exp}}(s'), && s \notin G,
\end{aligned}$$

and

$$\begin{aligned}
v^o_{\mathrm{exp}}(s) &= v_{\mathrm{exp}}(s), && s \in T_o,\\
v^o_{\mathrm{exp}}(s) &= \max_{\substack{a \in A_s \\ \mathrm{succ}(s,a) \subseteq S_o}} \sum_{s' \in S} \bar{P}(s'|s,a) v^o_{\mathrm{exp}}(s'), && s \in S_o \setminus T_o.
\end{aligned}$$



(a) T53′

(b) T55′

(c) TD43′

(d) TD45′

(e) D53′

(f) D55′

(g) DB43′

(h) DB45′

(i) B51′

Figure 3.14: Options obtained from pre-options using hierarchical dynamic programming

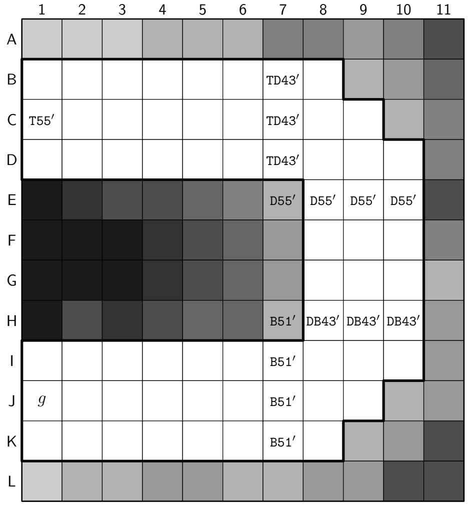


Figure 3.15: An "optimal" policy under the MER objective using hierarchical dynamic programming

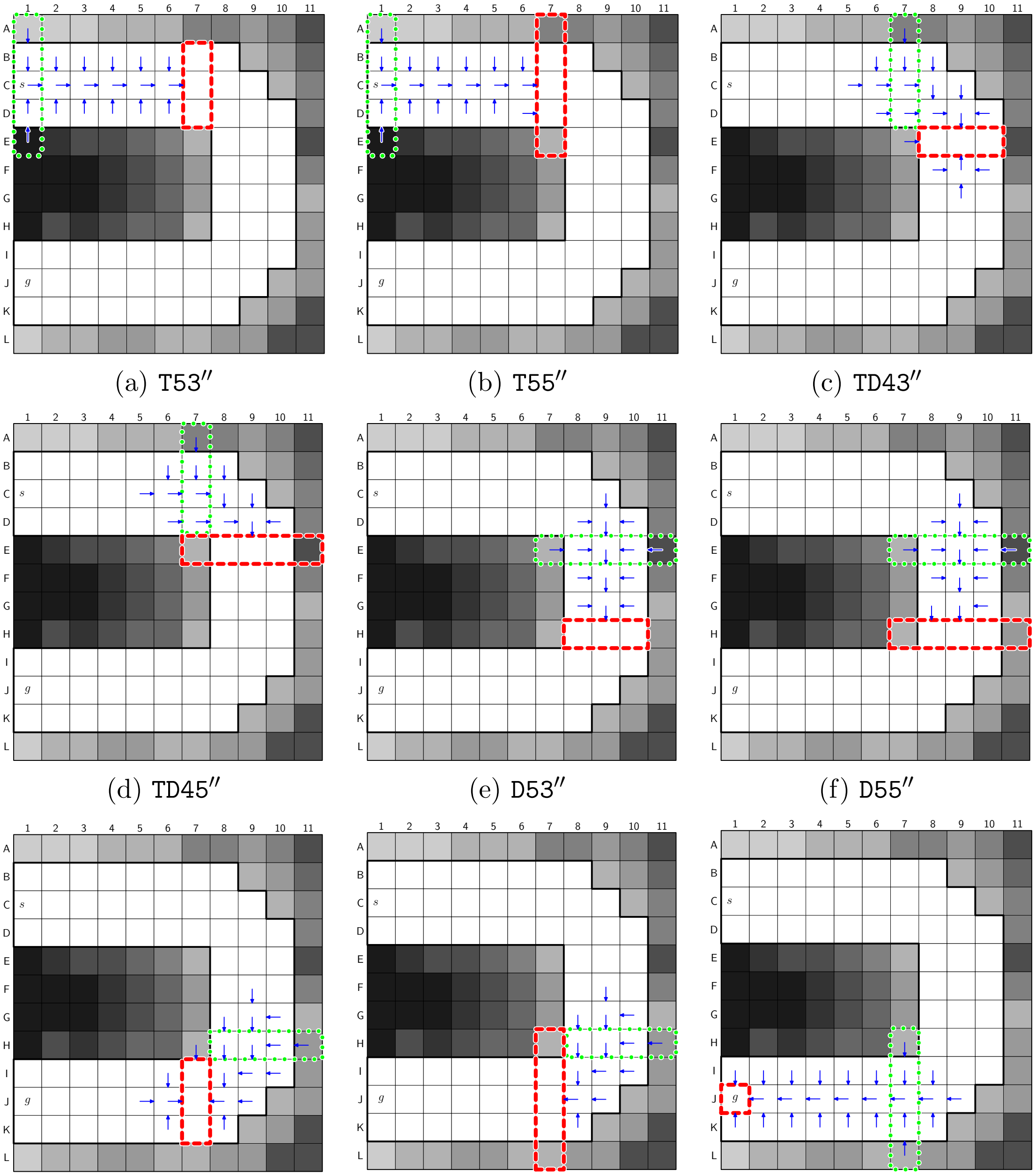
Using the pseudo-probability transformation, we can obtain an $\mathsf{MEU}_{\text{exp}}$ version of the hierarchical dynamic programming algorithm that solves this system of optimality equations, as long as there exists a recursively optimal policy with finite values.

For a risk-averse agent ($\gamma = 0.6$), Figure 3.16 shows the options obtained through the $\mathsf{MEU}_{\text{exp}}$ version of hierarchical dynamic programming and Figure 3.17 shows the recursively optimal policy using these options. Figure 3.18 and Figure 3.19 show the results for a risk-seeking agent ($\gamma = 2.0$).

3.6.3 Options as Temporally Extended Actions

Options (Sutton *et al.*, 1999b; Precup, 2000) are a general model of temporal abstractions for MDP models. Both action sequences and simple options are special cases of options. Options have been used in both DT planning and reinforcement learning to speed up planning or learning (Precup, 2000). Moreover, many other DT planning and reinforcement learning methods can be viewed as special cases of options, such as state space decomposition methods (Dean and Lin, 1995), HAM (hierarchies of abstract machines) (Parr and Russell, 1998; Parr, 1998), and MAXQ (Dietterich, 2000).

An option is a partial policy with a set of initial states and a stochastic termination condition. An option can be initiated in an initial state. Once initiated, it follows the partial



(a) T53″

(b) T55″

(c) TD43″

(d) TD45″

(e) D53″

(f) D55″

(g) DB43″

(h) DB45″

(i) B51″

Figure 3.16: Options obtained from pre-options for a risk-averse agent using hierarchical dynamic programming

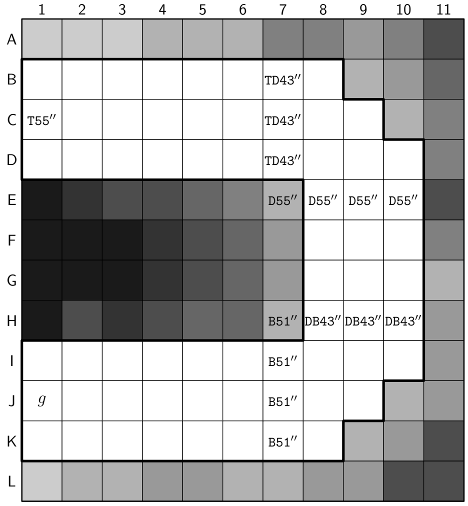


Figure 3.17: An "optimal" policy under the $\mathsf{MEU}_{\text{exp}}$ objective for a risk-averse agent using hierarchical dynamic programming

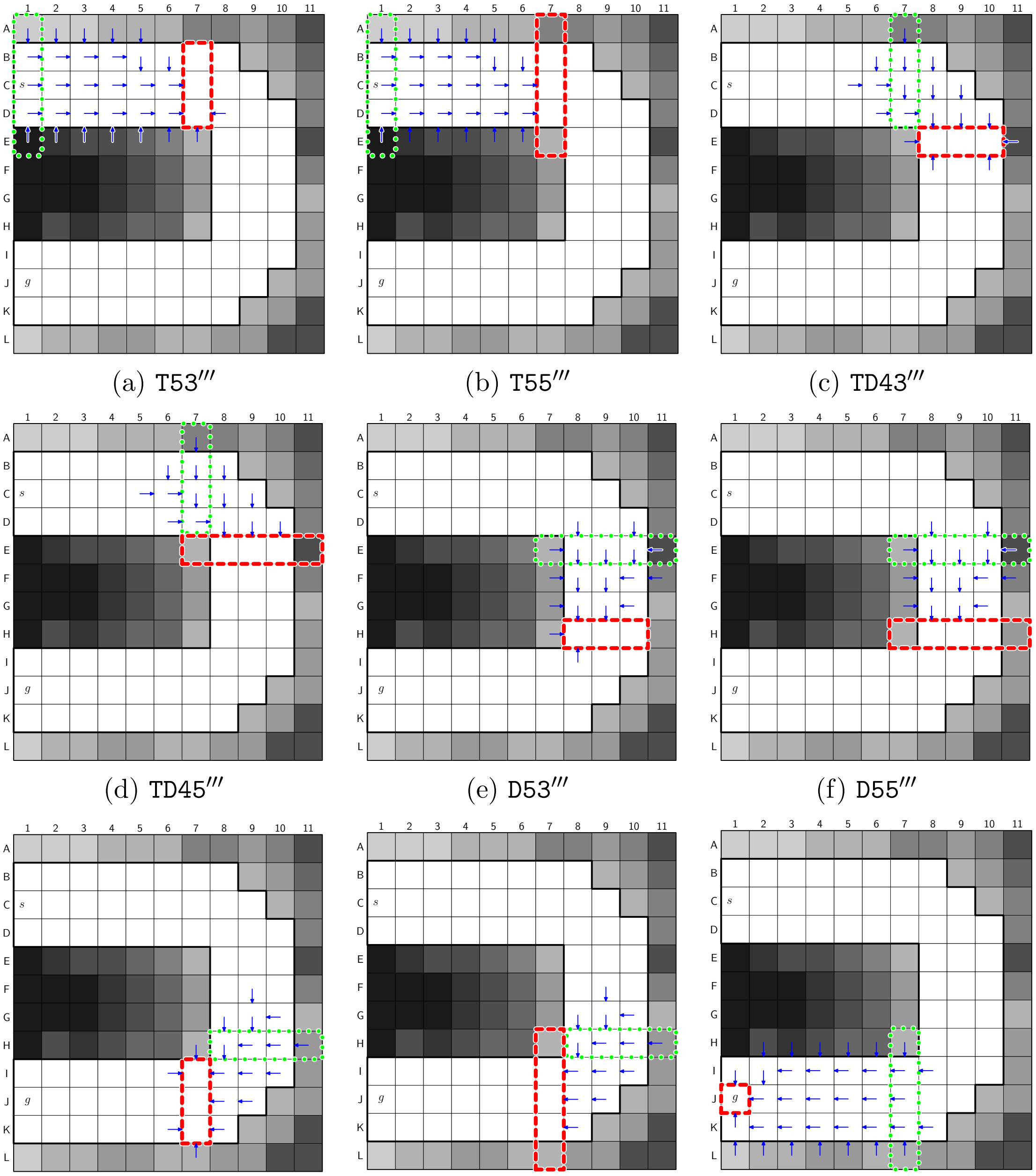
policy until the termination condition holds. Therefore, an option can also be viewed as a temporally extended action, whose state transition probabilities are derived from the partial policy and the termination condition.

To formally define options, we need the concept of partial histories. A partial history $h_{(t)} = (s_{(0)}, a_{(0)}, \ldots, s_{(t)})$ is the sequence of all states and actions of length $2t+1$ such that there exists $h_T = (s_0, a_0, \ldots, s_T) \in H_T$ where $T \geq t$ and $h_{(t)}$ is a suffix of h_T, that is, $s_{T-t+k} = s_{(k)}$ and $a_{T-t+k} = a_{(k)}$ for all $0 \leq k < t$ as well as $s_T = s_{(t)}$. In other words, it is

$$h_T = (s_0, a_0, \ldots, s_{T-t-1}, a_{T-t-1}, \underbrace{s_{T-t}, a_{T-t}, \ldots, s_T}_{h_{(t)}}).$$

The partial history $h_{(0)} = (s_{(0)})$ can be simplified to $s_{(0)}$. We denote the set of all partial histories of all lengths as (H).

An option (Precup, 2000) $o : (H) \mapsto \mathcal{P}(A \cup \{a_\tau\})$ is a mapping from partial policies to probability distributions over all actions and a special action a_τ that denotes the termination of the option, and $o(h_{(t)}, a)$ denotes the probability of performing action a when the partial history since o was initiated is $h_{(t)}$. In other words, an option is a partial-history-dependent policy. We also require that $o(s, a_\tau)$ can only be zero or one: $o(s, a_\tau) = 1$ means that o cannot be initiated in state s, and $o(s, a_\tau) = 0$ means that o can be initiated in state s. Once o is initiated, the agent follows the probabilities $o(h_{(h)}, a)$ to choose an action



(a) T53$'''$ (b) T55$'''$ (c) TD43$'''$

(d) TD45$'''$ (e) D53$'''$ (f) D55$'''$

(g) DB43$'''$ (h) DB45$'''$ (i) B51$'''$

Figure 3.18: Options obtained from pre-options for a risk-seeking agent using hierarchical dynamic programming

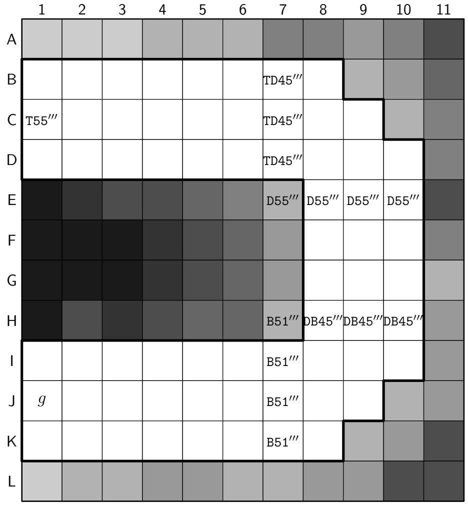


Figure 3.19: An "optimal" policy under the $\mathsf{MEU}_{\text{exp}}$ objective for a risk-seeking agent using hierarchical dynamic programming

to perform, and the option is terminated when the termination action $a_\mathcal{T}$ is chosen. For planning problems with goal states, it is therefore reasonable to continue to assume that options terminate with probability one.

An action sequence is an option that terminates after a fixed number of actions, and the choice of actions is deterministic for each decision epoch. A simple option is an option where the partial-history-dependent policy is stationary and the termination condition is a set of termination states.

Options are also similar to regular actions. Recall that a regular action is fully specified by the states in which it can be performed, the transition probabilities, and the rewards associated with the state transitions. We have mentioned that the set of states in which an option can be initiated is determined by the values of $o(s, a_\mathcal{T})$, which can only be zero or one. An option o also defines transition probability distributions $P(\cdot|s, o)$ over the state space, such that $P(s'|s, o)$ is the probability of terminating in state s' when option o was initiated in state s. On the other hand, the total reward received from the initiation of an option until the termination of the option is not a function of the initial state, the termination state, and the option itself. The total rewards rather form a probability distribution. For the MER objective, we can define the reward function as $r(s, o, s') = E^{s,o}[w_{(\mathcal{T})}|s_{(\mathcal{T})} = s']$, according to our discussion of a proper reward model for the MER objective in Section 2.3, where τ is the random variable indicating the number of decision epochs before the option terminates. For the $\mathsf{MEU}_{\text{exp}}$ objective, we can instead define the reward function as $r(s, o, s') = \log_\gamma E^{s,o}[\gamma^{w_{(\tau)}}|s_{(\tau)} = s']$, according to our discussion in Section 3.2. Then we can treat options as regular actions, and an MDP with options is equivalent to another MDP, whose exact definition is dependent on the planning objective, namely, either the MER objective or the $\mathsf{MEU}_{\text{exp}}$ objective.

With the above construction of transition probabilities and the reward function, we can define pseudo-probabilities for options as

$$\begin{aligned}\bar{P}(s'|s, o) = P(s'|s, o)\gamma^{r(s,o,s')} &= P(s'|s, o)E^{s,o}\left[\gamma^{\sum_{t=0}^{\tau-1} r_{(t)}}\middle| s_{(\tau)} = s'\right] \\ &= P(s'|s, o)E^{s,o}\left[\gamma^{w_{(\tau)}}|\, s_{(\tau)} = s'\right]. \end{aligned} \tag{3.34}$$

The pseudo-probability transformation then replaces probabilities $P(s'|s, o)$ with pseudo-probabilities $\bar{P}(s'|s, o)$. For specific forms of options, it is often more convenient to determine the relationship between the transition probabilities for the option $P(s'|s, o)$ and the transition probabilities for primitive actions $P(s'|s, a)$, and replace $P(s'|s, a)$ with $\bar{P}(s'|s, a)$ in this relationship to obtain a definition for $\bar{P}(s'|s, o)$. Then we need to prove that the definition satisfies Eq. (3.34). This is exactly what we did for action sequences in Section 3.6.1 and for simple options in Section 3.6.2.

3.6.4 Pseudo-Discount Factor Transformation?

The question remains whether we can also use the pseudo-discount factor transformation to obtain the same generalization. Recall that in the simplest case of the pseudo-discount factor transformation, we need to replace $\beta(s, a, s')$ with $\gamma^{r(s,a,s')}$. Now we consider options, so we need to replace $\beta(s, o, s')$ with $\gamma^{r(s,o,s')}$. However, the transition from s to s' that resulted from an option does not correspond to a single value $r(s, o, s')$, but a distribution of possible rewards collected along the way. Therefore, a direct transformation is not possible. But in fact, as mentioned in Section 3.2, the reward model $r(s, a, s')$ is just a simplification of a complete reward model where the rewards follow a nondegenerate distribution. Therefore, we could start with an MDP solution method under the MER_β objective using such a general reward definition, find the correspondences, and do the transformation accordingly. This is overly complex for our purpose.

3.7 Pseudo-Discount Factor Transformation and Factored Probabilities

Factored probabilities in factored MDPs are another common type of implicitly represented probabilities. For planners using factored probabilities, I suggest the pseudo-discount factor transformation, since it is inconvenient to use the pseudo-probability transformation (see Section 3.7.1.2). In this section, I show that the pseudo-discount factor transformation can be conveniently applied to structural dynamic programming methods such as SPUDD (Hoey *et al.*, 1999).

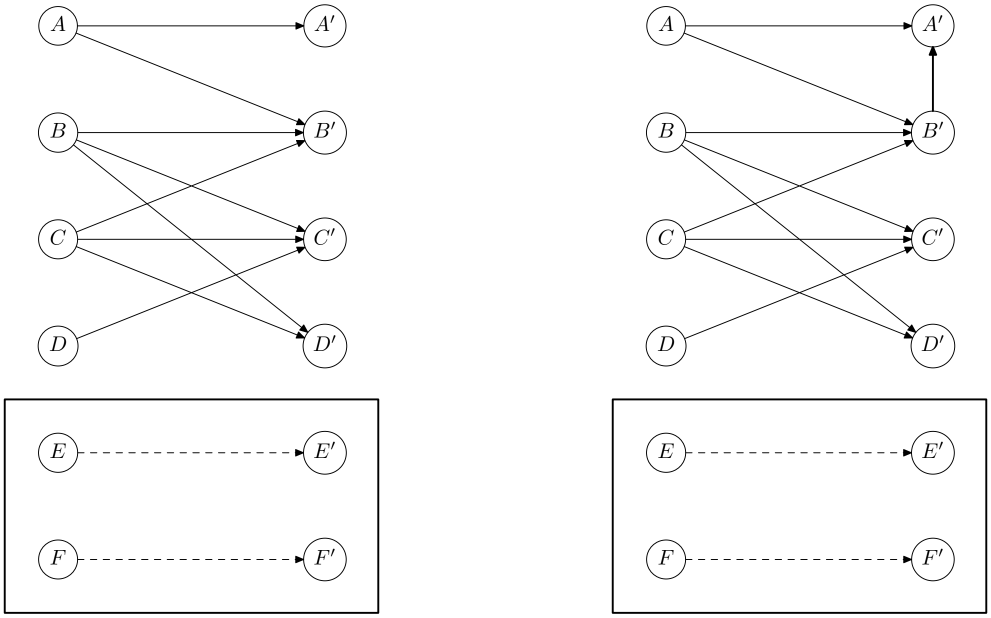
Section 3.7.1 introduces the factored representation of MDPs, and Section 3.7.2 discusses the SPUDD method for the MER_β objective. In Section 3.7.2.4, we discuss how to apply the pseudo-discount factor transformation to plan for the $\mathsf{MEU}_{\text{exp}}$ objective. We also use the painted-blocks problem to illustrate how the SPUDD method works.

3.7.1 Factored Representation of MDPs

In this section, we first give a brief overview of the factored representation of MDPs. A factored MDP is based on the factored representation of states. In AI planning problems, it is common to represent states using a set of n features, or factors[5] $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ such that $S = \Omega_{X_1} \times \Omega_{X_2} \times \cdots \times \Omega_{X_n}$, where Ω_{X_i} is the set of possible values of the factor X_i for $i = 1, 2, \ldots, n$. Consequently, a state is represented as a vector of factor values $s = (x_1, x_2, \ldots, x_n)$, where $x_i \in \Omega_{X_i}$ for $i = 1, 2, \ldots, n$. Therefore, a factored representation does not need to enumerate all states, but only needs to enumerate the factors and the values for each factor. In contrast, we refer to the representation that enumerates all states as the flat representation.

For factored MDPs, it is often assumed that the action set is the same for all states, that is, $A_s = A$ for all $s \in S$, and all actions can be enumerated. Hence for each action,

[5] We use the following notational convention: $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ for collections of sets of factors; $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ for sets of factors; X, Y, Z for individual factors; x, y, z for values of factors; and $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ for vectors of such values. These symbols can also have subscripts, which are always in a regular math font.



(a) A simple 2TBN

(b) A 2TBN with a synchronic arc

A	B'	$P(A'\|A, B'; a)$
T	T	0.9
T	F	0.5
F	T	0.0
F	F	0.5

A	B	C	$P(B'\|A, B, C; a)$
T	T	T	0.9
T	T	F	0.9
T	F	T	0.9
T	F	F	0.1
F	T	T	1.0
F	T	F	1.0
F	F	T	0.0
F	F	F	0.0

B	C	D	$P(C'\|B, C, D; a)$
T	T	T	0.9
T	T	F	0.0
T	F	T	0.0
T	F	F	0.1
F	T	T	0.5
F	T	F	0.0
F	F	T	0.5
F	F	F	0.0

B	C	$P(D'\|B, C; a)$
T	T	0.8
T	F	1.0
F	T	0.8
F	F	0.0

(c) Factored CPTs for the 2TBN in (b)

Figure 3.20: 2-time dynamic Bayesian networks

we can specify the transition probabilities between any pair of states. Using the factored representation, the transition probabilities of actions can then be represented as 2-stage temporal Bayesian networks (2TBNs) (Dean and Kanazawa, 1989; Boutilier *et al.*, 1999). Figure 3.20 shows some examples of 2TBNs. A 2TBN is a directed acyclic graph (dag) consisting of two layers of nodes, where the first layer consists of factors at the current time (denoted as X_i), and the second layer consists of the factors at the next time (denoted as X'_i). Because of the 1-1 correspondence between nodes and factors, we refer to the nodes simply as factors. Since we assume that the current state is always observable, all current-time factors are observable and it is sufficient that only the next-time factors have parents. For the next-time factor X'_i, we use $\boldsymbol{\Pi}'_i$ to denote the set of its parent factors. For simple 2TBNs, there are only directed arcs from the first layer to the second layer, implying the next-time factors are uncorrelated (Boutilier *et al.*, 1999). It is, however, common for the next-time factors to be correlated. In this case, we need to have directed arcs among nodes in the second layer. Such arcs are called synchronic (Boutilier *et al.*, 1999). Figure 3.20(a) is a simple 2TBN and Figure 3.20(b) is a 2TBN with a synchronic arc $B' \to A'$. In both networks, the nodes concerning factors E and F are linked with dashed lines, indicating that these factors are not affected by the corresponding action. Such factors can often be omitted in the 2TBN representation.

For a given action a, instead of explicitly representing the transition probabilities as $P(s'|s,a) = P(x'_1, x'_2, \ldots, x'_n|x_1, x_2, \ldots, x_n; a)$,[6] we have a set of factored conditional probability tables (CPTs) $P\big(X'_i\big|\boldsymbol{\Pi}'_i(a), a\big)$, where $\boldsymbol{\Pi}'_i(a)$ indicates the set of parent nodes of X'_i in the 2TBN for action a. The collection of all factored CPTs form an implicit representation of the transition probabilities, since from the theory of Bayesian networks (Pearl, 1988), it holds that

$$P(s'|s,a) = P(x'_1, x'_2, \ldots, x'_n|x_1, x_2, \ldots, x_n; a) = \prod_{i=1}^{n} P\big(x'_i\big|\boldsymbol{\pi}'_i(a), a\big), \tag{3.35}$$

where $\boldsymbol{\pi}'_i(a)$ indicates a vector of values for $\boldsymbol{\Pi}'_i(a)$. Figure 3.20(c) shows the factored CPTs for the 2TBN shown in Figure 3.20(b). In the example, we assume the variables have

[6] For probabilities and reward functions involving values of factors, we use semicolons in the notation to distinguish the values of factors and the action.

binary values: T(rue) and F(alse). Following the convention of Bayesian networks, we do not list probabilities such as $P(\neg A'|A, B'; a)$ since $P(\neg A'|A, B'; a) = 1 - P(A'|A, B'; a)$. The unaffected factors E and F are not shown, since the respective factored CPTs are identity transition probability tables, that is, $P(E'|E; a) = 1$ if and only if $E' = E$, for example.

Since an action often only affects a small number of factors, the number of factors in $\boldsymbol{\Pi}'_i(a)$, n_a, is often much smaller than the number of all factors n. Therefore for each action, the number of total entries in all relevant CPTs (for example, 2^{n_a} if the values are binary) is much smaller than the number of total entries in a complete enumeration of the transition probability table (for example, 2^n if the values are binary). Therefore, the factored representation avoids enumerating the state space, thus is (approximately) logarithmically more compact than the flat representation (Boutilier *et al.*, 1999).

To summarize, a factored MDP M is a 4-tuple $(\boldsymbol{X}, A, P, r)$, where

- $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ is the set of factors such that $S = \Omega_{X_1} \times \Omega_{X_2} \times \cdots \times \Omega_{X_n}$;
- A is the common action space for all states;
- P is the set of factored CPTs $P\big(X_i \big| \boldsymbol{\Pi}'_i(a), a\big)$ for all $i = 1, 2, \ldots, n$ and all $a \in A$;
- r is a reward function $r(x_1, x_2, \ldots, x_n; a; x'_1, x'_2, \ldots, x'_n)$ for all $a \in A$ and all $x_i, x'_i \in \Omega_{X_i}$ where $i = 1, 2, \ldots, n$.[7]

Existing algorithms for factored MDPs use the reward model $r(s, a)$ since they are planning under risk-neutral objectives. We instead use the reward model $r(s, a, s')$ as required by the $\mathsf{MEU}_{\text{exp}}$ objectives. Therefore, we modify the original MER algorithms to reflect this change.

3.7.1.1 The Painted-Blocks Example

The painted-blocks problem can have a feature-based representation known as the probabilistic STRIPS operators (PSO) representation (Fikes and Nilsson, 1971; Boutilier *et al.*,

[7] The reward function may also have compact representations, for example, using ADDs (Hoey *et al.*, 1999) or linear functions (Koller and Parr, 2000). These representations are quite different and there is no known unified representation. For now, we use this generic format to avoid unnecessary details.

1999). In this representation, however, we need to distinguish individual blocks by naming them. For each blocks, there are several features, and for each feature, there are a finite number of possible values. More concretely, for each block `X`, we have the following features (predicates):

- `(X_on Y)`, where `Y` is either `table` or another block different from `X`,[8]
- `(X_clear YN)`, where `YN` is either `yes` or `no`, and
- `(X_color C)`, where `C` is either `black` or `white`.

For a problem with n blocks, there are $3n$ factors in this representation. Figure 3.21 shows the named version of the problem from Figure 1.6, where the names of blocks are shown for the initial state, but not for the goal state, which would require enumerating 60 ($= 5 \times 4 \times 3$) different configurations, so we do not label the blocks. However, it is worth noting that not all combinations of values form a valid state in the original problem, since these factors are correlated. For example, the following combination is not a valid state for the problem from Figure 3.21

```
((A_on B)     (A_clear yes) (A_color black)
 (B_on table) (B_clear yes) (B_color white)
 (C_on B)     (C_clear yes) (C_color black)
 (D_on E)     (D_clear yes) (D_color white)
 (E_on D)     (E_clear yes) (E_color white)),
```

since two blocks cannot be on top of the same block, and a block is clear if and only if no other block is on top of it. We refer to such combinations as invalid states.

The description of an action consists of a precondition list and a set of possible outcomes associated with their respective probabilities, where an outcome is a list of predicates that change values. If no predicates change their values in an outcome, this outcome can be omitted, and its probability is the complement of the sum of all other outcomes (see action

[8] Traditionally, the blocks-world problem is represented with a `X_on_table` predicate with binary values `yes` and `no`. Here we combine the "on-table" and "on-block" predicates into a single predicate with multiple values, in order to make the representation compact for illustration purposes.

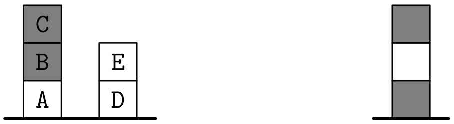


Figure 3.21: The named version of the painted-blocks problem from Figure 1.6

```
(move_X_from_Y_to_Z)
pre:     ((X_on Y) (X_clear yes) (Y_clear no) (Z_clear yes))
outcome: (0.5 (X_on Z) (Y_clear yes) (Z_clear no))
         (0.5 (X_on table) (Y_clear yes))

(move_X_from_table_to_Z)
pre:     ((X_on table) (X_clear yes) (Z_clear yes))
outcome: (0.5 (X_on Z) (Z_clear no))

(move_X_from_Y_to_table)
pre:     ((X_on Y) (X_clear yes) (Y_clear no))
outcome: (1.0 (X_on table) (Y_clear yes))

(paint_X_C)
pre:     ()
outcome: (1.0 (X_color C))
```

Figure 3.22: Actions for the painted-blocks problem

move_X_from_table_to_Z in Figure 3.22). The complete description of all possible actions for the painted-blocks problem is shown in Figure 3.22.[9] For this problem, X, Y, and Z can be any of A, B, C, D, or E, as long as there is no duplication.

Boutilier *et al.* (1999) discussed the equivalence of the PSO and 2TBN representations, and showed how to convert the 2TBN representation to the PSO representation. We now use the painted-blocks domain to show how to convert the PSO representation to the 2TBN representation, and then in Section 3.7.3 show how to solve painted-blocks problems using the 2TBN representation.

Since a 2TBN represents the transition probabilities for all states but a PSO uses a precondition list to restrict its applicability in different states, we need to extend the definition of a PSO action to states where it is not applicable. We simply define that the action

[9] We include (Y_clear no) in the preconditions for actions move_X_from_Y_to_Z and move_X_from_Y_to_table, since (X_on Y) implies (Y_clear no) in a valid state. Without (Y_clear no), these actions are also applicable to invalid states. We include this predicate for clarity, but the algorithms still work without it, since the agent cannot enter an invalid state anyway.

results in self-looping in those states. We also define the immediate rewards for self-loops to be a pure cost (-1, for example), so that self-looping results in an infinite total cost and thus this action will not be preferred in planning.

We now illustrate the conversion using action move_X_from_Y_to_Z. First, those factors not appearing in the precondition or outcome lists are irrelevant, and their values will not be changed. Therefore, we only need to consider factors X_on, X_clear, Y_clear, and Z_clear. Since the factors in each outcome are correlated, we need to use 2TBNs with synchronic arcs. We can consider a hypothetical next-time node Outcome, whose values are the different outcomes (success and failure for action move_X_from_Y_to_Z) plus a special value inapplicable. The 2TBN has links from all factors in the precondition list to the hypothetical Outcome node, and then links from node Outcome to real next-time factors that appear in the outcome list, namely, X_on, Y_clear, and Z_clear. The structure of the 2TBN is illustrated in Figure 3.23, where there are also arcs directly from the current-time factors to the next-time factors for a reason to be discussed next. The CPT for node Outcome is straightforward: if the precondition is not satisfied, the value is inapplicable with certainty; otherwise, the value can be any of the outcomes with its respective probability. The CPTs for nodes in the outcome lists are also simple: if the value for Outcome is inapplicable, the next-time factors do not change values; otherwise, a next-time factor changes its value deterministically according to the outcome it belongs to (or it does not change value if an outcome does not include it). More concretely, the CPTs for action move_X_from_Y_to_Z are shown in Table 3.6, where the action move_X_from_Y_to_Z and irrelevant factors and values are omitted to be concise. Notice that in order to model "unchanged values", we need to include values of current-time nodes in the CPTs for next-time nodes. The resulting 2TBN has an additional next-time node Outcome, and thus has synchronic arcs starting from node Outcome.

We can also eliminate the hypothetical next-time node Outcome in the 2TBN, which reduces the complexity of the internal representations of the planning methods to be discussed next. Suppose that we use the order of the "real" next-time nodes shown in Figure 3.23

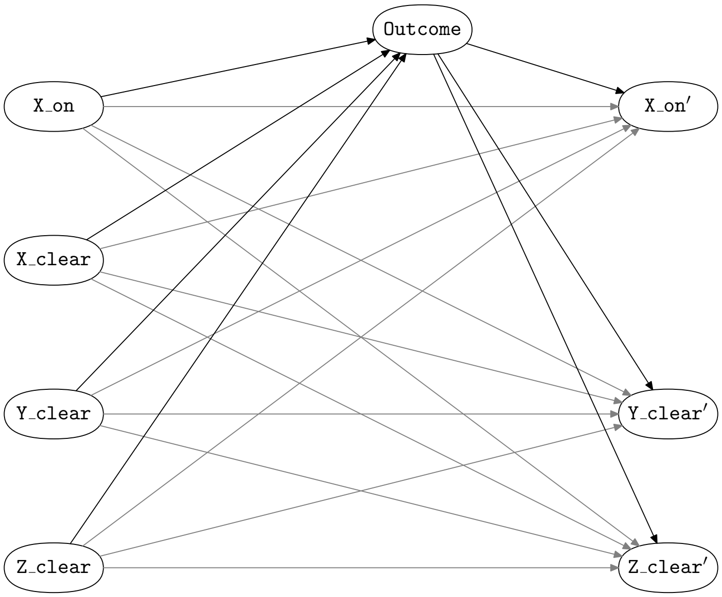


Figure 3.23: Convert move_X_from_Y_to_Z from a PSO representation to a 2TBN representation

Table 3.6: CPTs for move_X_from_Y_to_Z with the Outcome node

X_on	X_clear	Y_clear	Z_clear	$P(\texttt{Outcome}\|\cdots)$		
				success	failure	inapplicable
Y	yes	no	yes	0.5	0.5	0.0
all other combinations				0.0	0.0	1.0

X_on	X_clear	Y_clear	Z_clear	Outcome	$P(\texttt{X\_on}'\|\cdots)$		
					table	Z	···
Y	yes	no	yes	success	0.0	1.0	0.0
Y	yes	no	yes	failure	1.0	0.0	0.0
all other combinations					1.0 if $\texttt{X\_on} = \texttt{X\_on}'$ 0.0 if $\texttt{X\_on} \neq \texttt{X\_on}'$		

X_on	X_clear	Y_clear	Z_clear	Outcome	$P(\texttt{Y\_clear}'\|\cdots)$	
					yes	no
Y	yes	no	yes	success	1.0	0.0
Y	yes	no	yes	failure	1.0	0.0
all other combinations					1.0 if $\texttt{Y\_clear} = \texttt{Y\_clear}'$ 0.0 if $\texttt{Y\_clear} \neq \texttt{Y\_clear}'$	

X_on	X_clear	Y_clear	Z_clear	Outcome	$P(\texttt{Z\_clear}'\|\cdots)$	
					yes	no
Y	yes	no	yes	success	0.0	1.0
Y	yes	no	yes	failure	1.0	0.0
all other combinations					1.0 if $\texttt{Z\_clear} = \texttt{Z\_clear}'$ 0.0 if $\texttt{Z\_clear} \neq \texttt{Z\_clear}'$	

from top to bottom. Notice that we have

$$\begin{aligned}
&P\big(\texttt{X\_on}', \texttt{Y\_clear}', \texttt{Z\_clear}' \bigm| \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}\big)\\
&= P\big(\texttt{X\_on}' \bigm| \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}\big)\\
&\quad \cdot P\big(\texttt{Y\_clear}' \bigm| \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}'\big)\\
&\quad \cdot P\big(\texttt{Z\_clear}' \bigm| \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}'\big).
\end{aligned}$$

That is, eliminating Outcome introduces direct correlations among next-time factors X_on’, Y_clear’, and Y_clear’. Let $\Omega_{\texttt{Outcome}} = \{\texttt{success}, \texttt{failure}\}$ and $\Omega'_{\texttt{Outcome}} = \Omega_{\texttt{Outcome}} \cup \{\texttt{inapplicable}\}$. According to Figure 3.23, it holds that

$$\begin{aligned}
&P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{X\_on}', o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&= \sum_{o \in \Omega_{\texttt{Outcome}}} P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&\quad + P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{inapplicable})\\
&\qquad\quad \cdot P(\texttt{inapplicable}|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}).
\end{aligned}$$

If the values of the current-time features are such that the precondition holds, then it holds that

$$P(\texttt{inapplicable}|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}) = 0,$$

and

$$\begin{aligned}
&P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&= \sum_{o \in \Omega_{\texttt{Outcome}}} P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})\\
&= \sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{X\_on}' \in o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}),
\end{aligned}$$

where o also indicates the set of value assignments for the corresponding outcome and

$$\chi(A) = \begin{cases} 1, & A \text{ is True} \\ 0, & A \text{ is False} \end{cases}$$

for a predicate A. Otherwise, it holds that for all $o \in \Omega_{\texttt{Outcome}}$,

$$P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}) = 0.$$

It also holds that

$$P(\texttt{inapplicable}|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}) = 1$$

and

$$P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{inapplicable}) = \chi(\texttt{X\_on}' = \texttt{X\_on}).$$

Therefore, it holds that

$$P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}) = \chi(\texttt{X\_on}' = \texttt{X\_on}).$$

Similarly, we have

$$\begin{aligned}
&P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}') \\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Y\_clear}', o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}') \\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', o) \\
&\qquad \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}') \\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \\
&\qquad \cdot \frac{P(o, \texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})} \\
&= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \\
&\qquad \cdot \frac{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}.
\end{aligned}$$

If the values of the current-time features are such that the precondition holds, then it holds that

$$\begin{aligned}
&P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}') \\
&= \sum_{o \in \Omega_{\texttt{Outcome}}} P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \\
&\qquad \cdot \frac{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}
\end{aligned}$$

$$= \frac{1}{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}$$
$$\cdot \sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{Y\_clear}' \in o) \cdot \chi(\texttt{X\_on}' \in o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})$$
$$= \frac{\sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{X\_on}', \texttt{Y\_clear}' \in o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{\sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{X\_on}' \in o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}.$$

Otherwise, it holds that

$$P(\texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}')$$
$$= \frac{\chi(\texttt{Y\_clear}' = \texttt{Y\_clear}) \cdot \chi(\texttt{X\_on}' = \texttt{X\_on})}{P(\texttt{X\_on}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})} = \chi(\texttt{Y\_clear}' = \texttt{Y\_clear}).$$

We also have

$$P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}')$$
$$= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Z\_clear}', o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}')$$
$$= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}', o)$$
$$\cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}')$$
$$= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o)$$
$$\cdot \frac{P(o, \texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}$$
$$= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o)$$
$$\cdot \frac{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}.$$

If the values of the current-time features are such that the precondition holds, then it holds that

$$P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}')$$
$$= \sum_{o \in \Omega'_{\texttt{Outcome}}} P(\texttt{Z\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o)$$
$$\cdot \frac{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}$$
$$= \frac{1}{P(\texttt{X\_on}', \texttt{Y\_clear}'|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}$$
$$\cdot \sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{Z\_clear}' \in o) \cdot \chi(\texttt{X\_on}', \texttt{Y\_clear}' \in o) \cdot P(o|\texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})$$

Table 3.7: CPTs for move_X_from_Y_to_Z without the Outcome node

X_on	X_clear	Y_clear	Z_clear	$P(\texttt{X\_on}' \mid \cdots)$		
				table	Z	···
Y	yes	no	yes	0.5	0.5	0.0
all other combinations				1.0 if $\texttt{X\_on} = \texttt{X\_on}'$ 0.0 if $\texttt{X\_on} \neq \texttt{X\_on}'$		

X_on	X_clear	Y_clear	Z_clear	X_on′	$P(\texttt{Y\_clear}' \mid \cdots)$	
					yes	no
Y	yes	no	yes	table	1.0	0.0
Y	yes	no	yes	Z	1.0	0.0
all other combinations					1.0 if $\texttt{Y\_clear} = \texttt{Y\_clear}'$ 0.0 if $\texttt{Y\_clear} \neq \texttt{Y\_clear}'$	

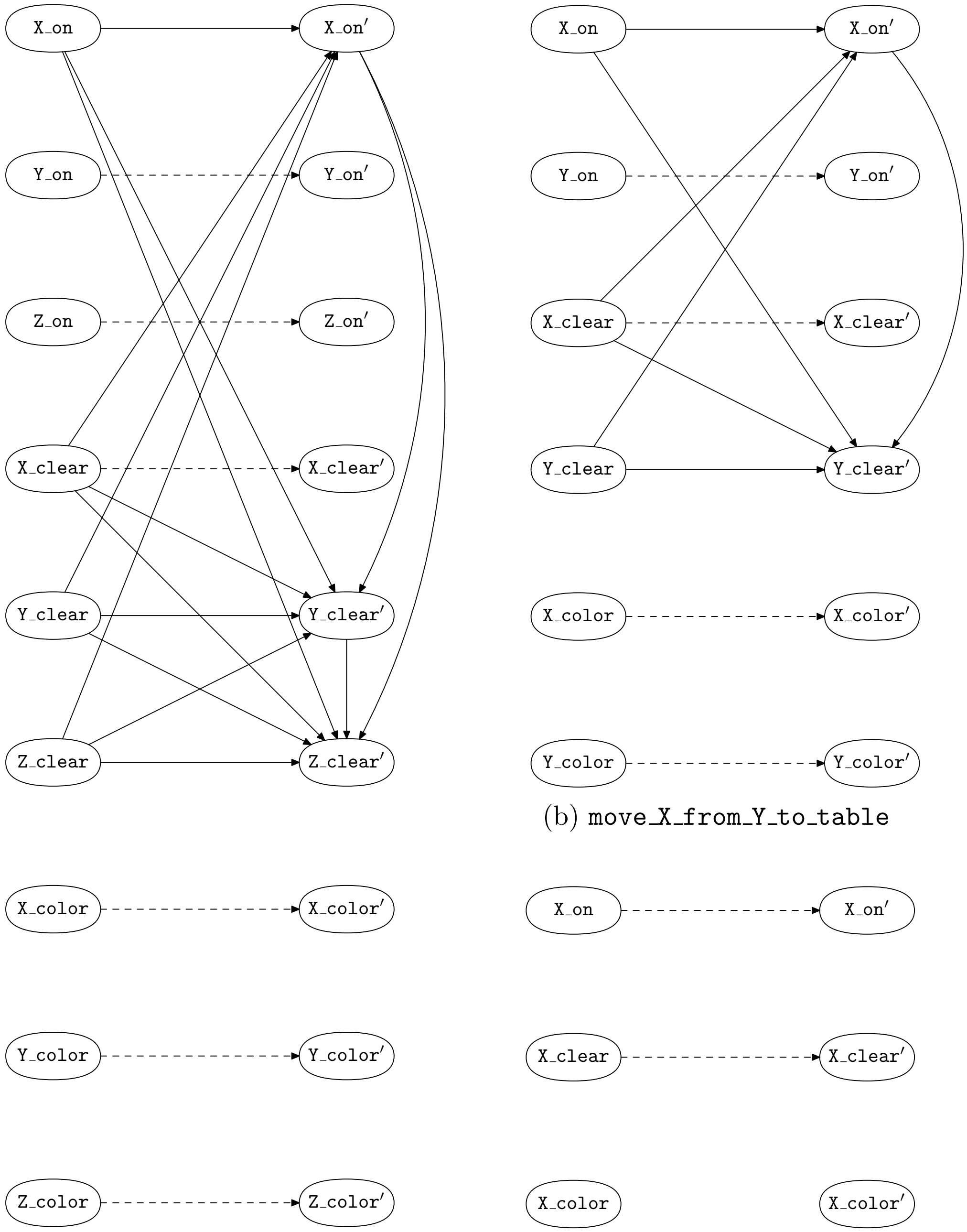
X_on	X_clear	Y_clear	Z_clear	X_on′	Y_clear′	$P(\texttt{Z\_clear}' \mid \cdots)$	
						yes	no
Y	yes	no	yes	table	yes	1.0	0.0
Y	yes	no	yes	Z	yes	0.0	1.0
all other combinations						1.0 if $\texttt{Z\_clear} = \texttt{Z\_clear}'$ 0.0 if $\texttt{Z\_clear} \neq \texttt{Z\_clear}'$	

$$= \frac{\displaystyle\sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{X\_on}', \texttt{Y\_clear}', \texttt{Z\_clear}' \in o) \cdot P(o \mid \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}{\displaystyle\sum_{o \in \Omega_{\texttt{Outcome}}} \chi(\texttt{X\_on}', \texttt{Y\_clear}' \in o) \cdot P(o \mid \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})}.$$

Otherwise, it holds that

$$\begin{aligned} & P(\texttt{Z\_clear}' \mid \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear}, \texttt{X\_on}', \texttt{Y\_clear}') \\ &= \frac{\chi(\texttt{Z\_clear}' = \texttt{Z\_clear}) \cdot \chi(\texttt{X\_on}' = \texttt{X\_on}, \texttt{Y\_clear}' = \texttt{Y\_clear})}{P(\texttt{X\_on}', \texttt{Y\_clear}' \mid \texttt{X\_on}, \texttt{X\_clear}, \texttt{Y\_clear}, \texttt{Z\_clear})} \\ &= \chi(\texttt{Z\_clear}' = \texttt{Z\_clear}). \end{aligned}$$

The above procedure is similar to eliminating node Outcome in the bucket elimination method (Dechter, 1996) with node Outcome having the highest index. The resulting 2TBN and CPTs will depend upon the ordering of the node in the bucket elimination procedure. Using this procedure, we can obtain 2TBNs representing actions for the painted-blocks domain, as shown in Figure 3.24, where only the features for relevant blocks are included to reduce the sizes of the networks. Notice that the resulting 2TBNs also have synchronic arcs to model the correlations of the next-time factors. The CPTs are shown in Table 3.7.



(b) `move_X_from_Y_to_table`

(a) `move_X_from_Y_to_Z`

(c) `paint_X_C`

Figure 3.24: 2TBN representation for actions of the painted-blocks domain

3.7.1.2 Difficulties with the Pseudo-Probability Transformation for Factored MDPs

For factored MDPs, the transition probabilities are represented implicitly as

$$P(s'|s,a) = \prod_{i=1}^{n} P\big(x'_i \big| \boldsymbol{\pi}'_i(a), a\big).$$

In the pseudo-probability transformation, the probabilities $P(s'|s,a)$ are replaced with the pseudo-probabilities $\bar{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')}$. Therefore, if we use the pseudo-probability transformation, the pseudo-probabilities for factored MDPs should be

$$\bar{P}(s'|s,a) = P(s'|s,a)\gamma^{r(s,a,s')} = \gamma^{r(s,a,s')} \prod_{i=1}^{n} P\big(x'_i \big| \boldsymbol{\pi}'_i(a), a\big). \tag{3.36}$$

However, there are two reasons why the pseudo-probability transformation is inconvenient for factored MDPs. First, planners using factored probabilities often manipulate implicit transition probabilities $P\big(x'_i \big| \boldsymbol{\pi}'_i(a), a\big)$ instead of explicit transition probabilities. Thus it is often difficult, if not impossible, to define implicit pseudo-probabilities in a meaningful way. In other words, the pseudo-probability transformation cannot deal with the factor $\gamma^{r(s,a,s')}$ in Eq. (3.36) directly. Second, planners using factored probabilities often perform some probabilistic reasoning, which may not be valid for pseudo-probabilities. Therefore, the pseudo-probability transformation is not suitable for planners using factored probabilities. On the other hand, the pseudo-discount factor transformation is more manageable for factored MDPs, as we will show for SPUDD.

3.7.2 SPUDD

The SPUDD (Stochastic Planning Using Decision Diagrams) method (Hoey *et al.*, 1999) is a generalization of the value iteration method for factored MDPs using algebraic decision diagrams. It is able to solve large-scale problems represented as factored MDPs, and has been generalized to approximate planning (St. Aubin *et al.*, 2000) and POMDPs (Hansen and Feng, 2000; Feng and Hansen, 2001). It has also been combined with heuristic search to solve much larger problems when the initial state is given (Feng and Hansen, 2002).

In this section, we first review the algebraic decision diagram (ADD) representation used by SPUDD in Section 3.7.2.1. Section 3.7.2.2 presents SPUDD in a reformulation exclusively

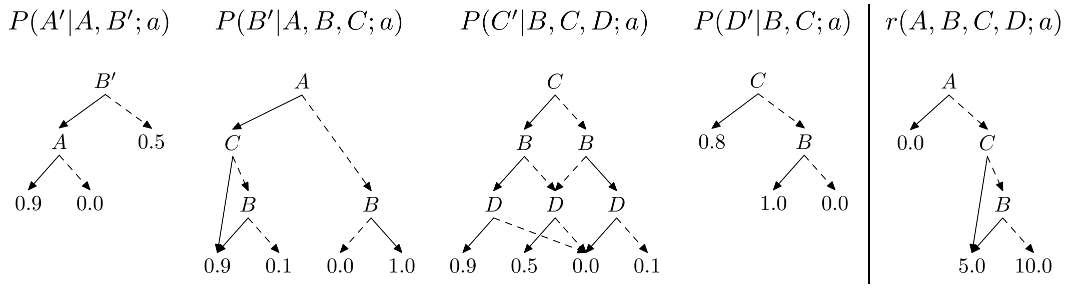


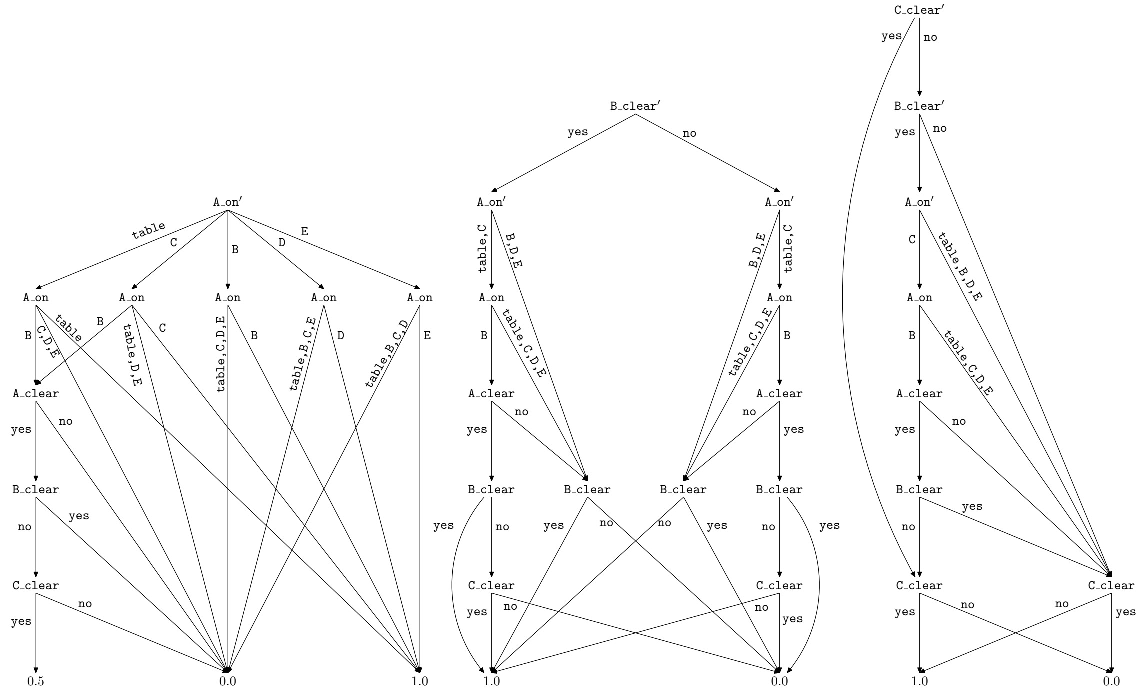
Figure 3.25: The ADD representation of factored CPTs from Figure 3.20(c) and the reward function

using ADD operations. This reformulation makes it easier to understand the SPUDD method, while the original formulation directly manipulates the ADD data structure and has many details irrelevant to our discussion. Section 3.7.2.3 discusses two extensions of SPUDD, in order to solve our painted-block example. One extension extends its applicability to 2TBNs with synchronic arcs, and the other one improves its efficiency.

3.7.2.1 Algebraic Decision Diagram Representation of Factored MDPs

It is possible to represent factored MDPs even more compactly using decision trees (Boutilier *et al.*, 1995, 2000) or algebraic decision diagrams (ADDs) (Hoey *et al.*, 1999). Since decision trees are special cases of ADDs and also less compact than ADDs, we focus on the representation using ADDs in our discussion. An ADD is a directed acyclic graph whose internal nodes are factors and whose leaf nodes are real numbers (Bahar *et al.*, 1993). An internal node has branches that correspond to its possible values. In general, an ADD represents a real-valued function on a finite discrete domain.[10] Figure 3.25 illustrates the ADD representation for the factored CPTs from Figure 3.20(c). Since all factors are binary in the example, the branches with a T value are drawn in solid lines and those with an F value in dashed lines. The reward function can also be represented in the same way, also shown in Figure 3.25. Here we assume that the reward function does not depend on the next-time

[10] Strictly speaking, ADDs only represent real-valued functions of boolean or binary variables. But in SPUDD 2.0 and later distributions, ADDs are also used for variables of an arbitrary but finite number of values. This is done by using $\lceil \log K \rceil$ boolean proxy variables for each variable, where K is the number of distinct values of the original variable. In the resulting ADDs, there can be spurious branches that do not correspond to meaningful values of the original variables. The values of these branches are set to zero, and all ADD operations that are relevant to dynamic programming can be carried out without altering the results.



(a) $P(\texttt{A\_on}'|\cdots)$ (a) $P(\texttt{B\_clear}'|\cdots)$ (a) $P(\texttt{C\_clear}'|\cdots)$

Figure 3.26: The ADD representation of factored CPTs for action `move_A_from_B_to_C`

state, and use the simpler notation $r(A, B, C, D; a)$. When necessary, we use subscripts to distinguish the domain of the function represented as an ADD. For example, the reward function can be denoted as $r_{\{A,B,C,D\}}(\cdot, a)$. Here we use $\cdot$ to emphasize that r is a function. When the function takes more than one argument, we use more dots such as : and $\vdots$, and use matching dots to indicate the same arguments in different functions.

The ADD representation of CPTs is also applicable to the painted-blocks problem. As an example, the CPTs for action `move_A_from_B_to_C` are converted to ADDs shown in Figure 3.26.

An ADD can be manipulated using ADD operations. Standard ADD operations are provided in ADD packages.[11] A binary decision diagram (BDD) is a 0-1 valued ADD for representing boolean functions (1 is True and 0 is False). There are a set of operations specialized for BDDs. A BDD is also used to represent a set of states S by representing its characteristic function χ_S, that is, $s \in S$ if and only if $\chi_S(s) = \text{T}$. For our purpose, the relevant ADD and BDD operations include:

[11] We use the CUDD package (Somenzi, 2004).

- Change of variables:[12] let $f_{\boldsymbol{X}}(\cdot)$ be an ADD on $\boldsymbol{X}$, and $\boldsymbol{Y}$ be a set of variables of the same cardinality as $\boldsymbol{X}$ such that $\boldsymbol{X} \cap \boldsymbol{Y} = \varnothing$ and $\Omega_{\boldsymbol{X}} = \Omega_{\boldsymbol{Y}}$,

  $$g_{\boldsymbol{Y}}(:) = f_{\boldsymbol{X}}(\cdot)[\boldsymbol{X}/\boldsymbol{Y}] \qquad \text{if and only if} \qquad g(\boldsymbol{y}) = f(\boldsymbol{x}),$$

  where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ are values of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively, and $x_i = y_i$ for all $i = 1, 2, \ldots, n$.

- Function composition:[13] let $f_{\boldsymbol{X}}(\cdot)$ be an ADD on $\boldsymbol{X}$, $X_i \in \boldsymbol{X}$, and $g_{\boldsymbol{X}}(\cdot)$ be an ADD on $\boldsymbol{X}$ whose values are in Ω_{X_i},

  $$h_{\boldsymbol{X}}(\cdot) = f_{\boldsymbol{X}}(\cdot)[X_i \to g_{\boldsymbol{X}}(\cdot)] \qquad \text{if and only if}$$
  $$h(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_{i-1}, g(\boldsymbol{x}), x_{i+1}, \ldots, x_n),$$

  where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$.

- Function inversion:[14] let $f_{\boldsymbol{X}}(\cdot)$ be an ADD on $\boldsymbol{X}$, and $y \in \mathbb{R}$,

  $$A_{\boldsymbol{X}} = f_{\boldsymbol{X}}^{-1}(y) \qquad \text{if and only if} \qquad \boldsymbol{x} \in A_{\boldsymbol{X}} \Leftrightarrow f(\boldsymbol{x}) = y,$$

  where $A_{\boldsymbol{X}}$ is a set represented as a BDD.

- Vector addition/subtraction/multiplication/maximization:[15] let $f_{\boldsymbol{X}}(\cdot), g_{\boldsymbol{X}}(\cdot)$ be ADDs on $\boldsymbol{X}$,

  $$h_{\boldsymbol{X}}(\cdot) = f_{\boldsymbol{X}}(\cdot) \circledcirc g_{\boldsymbol{X}}(\cdot) \qquad \text{if and only if} \qquad h(\boldsymbol{x}) = f(\boldsymbol{x}) \circ g(\boldsymbol{x}),$$

  where $\circledcirc$ is one of $\oplus, \ominus, \odot, \varovee$ and $\circ$ is one of $+, -, \cdot, \vee$, respectively. The $\vee$ operator is defined as $a \vee b = \max(a, b)$.

[12] Implemented as `Cudd_addSwapVariables` in the CUDD package.

[13] Implemented as `Cudd_addCompose` in the CUDD package.

[14] Implemented as `Cudd_addBddInterval` in the CUDD package with the interval containing only a single value.

[15] Implemented as `Cudd_addPlus`/`Cudd_addMinus`/`Cudd_addTimes`/`Cudd_addMaximum` in the CUDD package, respectively.

- Generalized matrix multiplication:[16] let $f_{\boldsymbol{X},\boldsymbol{Y}}(\cdot,:)$ and $g_{\boldsymbol{Y},\boldsymbol{Z}}(:,\vdots)$ be ADDs on $\boldsymbol{X}\cup\boldsymbol{Y}$ and $\boldsymbol{Y}\cup\boldsymbol{Z}$ respectively, where $\boldsymbol{X}\cap\boldsymbol{Y}=\boldsymbol{Y}\cap\boldsymbol{Z}=\varnothing$,

  $$h_{\boldsymbol{X},\boldsymbol{Z}}(\cdot,\vdots)=f_{\boldsymbol{X},\boldsymbol{Y}}(\cdot,:)\otimes_{\boldsymbol{Y}} g_{\boldsymbol{Y},\boldsymbol{Z}}(:,\vdots)\quad\text{if and only if}\quad h(\boldsymbol{x},\boldsymbol{z})=\sum_{\boldsymbol{y}\in\Omega_{\boldsymbol{Y}}} f(\boldsymbol{x},\boldsymbol{y})g(\boldsymbol{y},\boldsymbol{z}).$$

  If further $\boldsymbol{X}\cap\boldsymbol{Z}=\varnothing$, this operation is a regular matrix multiplication, where f is a $|\Omega_{\boldsymbol{X}}|\times|\Omega_{\boldsymbol{Y}}|$ matrix, g is a $|\Omega_{\boldsymbol{Y}}|\times|\Omega_{\boldsymbol{Z}}|$ matrix, and h is a $|\Omega_{\boldsymbol{X}}|\times|\Omega_{\boldsymbol{Z}}|$ matrix. It is also worth noting that $\boldsymbol{X}$ and $\boldsymbol{Z}$ can be empty sets, in which case the operation is reduced to a vector-matrix multiplication or a vector-vector multiplication (inner product).

- If-Then-Else (ITE):[17] let $g_{\boldsymbol{X}}(\cdot)$ and $h_{\boldsymbol{X}}(\cdot)$ be ADDs and $f_{\boldsymbol{X}}(\cdot)$ be a BDD,

  $$k_{\boldsymbol{X}}(\cdot)=\textsc{ite}\big(f_{\boldsymbol{X}}(\cdot),g_{\boldsymbol{X}}(\cdot),h_{\boldsymbol{X}}(\cdot)\big)\qquad\text{if and only if}\qquad k(\boldsymbol{x})=\begin{cases}g(\boldsymbol{x}), & \text{if } f(\boldsymbol{x}),\\ h(\boldsymbol{x}), & \text{if } \neg f(\boldsymbol{x}).\end{cases}$$

- Existential abstraction (or relational product):[18] let $f_{\boldsymbol{X},\boldsymbol{Y}}$ and $g_{\boldsymbol{X},\boldsymbol{Y}}$ be two BDDs,

  $$h_{\boldsymbol{X}}=f_{\boldsymbol{X},\boldsymbol{Y}}\bowtie_{\boldsymbol{Y}} g_{\boldsymbol{X},\boldsymbol{Y}}\quad\text{if and only if}\quad h_{\boldsymbol{X}}(\boldsymbol{x})=\exists\boldsymbol{y}_{\in\Omega_{\boldsymbol{Y}}}\big(f_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{x},\boldsymbol{y})\wedge g_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{x},\boldsymbol{y})\big).$$

  The result is also a BDD.

3.7.2.2 The SPUDD Method

The original SPUDD method (Hoey *et al.*, 1999) only deals with simple 2TBNs, where all next-time factors are uncorrelated. SPUDD uses carefully designed operators from ADD packages (Somenzi, 2004) to perform dynamic programming operations, and these operators can be interpreted as performing regression operations analogous to classical AI planning (Boutilier *et al.*, 2000). These operators make it possible that value functions and policies can also be compactly represented as ADDs, so that the value iteration steps can be

[16] Implemented as `Cudd_addMatrixMultiply` and `Cudd_addTimesPlus` in the CUDD package. They use different methods, and neither dominates the other in terms of time complexity. But for SPUDD, `Cudd_addMatrixMultiply` is faster empirically.

[17] Implemented as `Cudd_addIte` in the CUDD package.

[18] Implemented as `Cudd_bddExistAbstract` in the CUDD package.

performed repeatedly until the termination condition is met. In this section, we reformulate the original SPUDD using standard ADD operations exclusively. This reformulation helps us to extend the applicability of SPUDD and improve its efficiency, as to be shown in Section 3.7.2.3.

To take advantage of ADD operations, SPUDD splits the value update step into two steps with the help of q-functions. A q-function is similar to a value function but is a mapping from state-action pairs to real numbers. The SPUDD value update rule with q-functions for the MER_β objective is:

$$\begin{aligned} v^t(s) &= 0, && s \in G, \\ v^t(s) &= \max_{a \in A} q^t(s,a), && s \in S \setminus G, \\ q^t(s,a) &= \sum_{s' \in S} P(s'|s,a)\big(r(s,a,s') + \beta v^{t-1}(s')\big), && s \in S \setminus G, a \in A, \end{aligned} \tag{3.37}$$

where $\beta \in (0,1)$ is the discount factor.

A direct translation of the above procedure using ADDs is simple. We first multiply the factored CPTs together for each action to obtain the complete transition probability table

$$P_{\boldsymbol{X},\boldsymbol{X}'}(:|\cdot,a) = \bigodot_{i=1}^{n} P\big(X_i'\big|\boldsymbol{\Pi}_i'(a),a\big),$$

then use ADD operations to perform the value-update rule Eq. (3.37), where the expectation is realized using the generalized matrix multiplication operator:

$$q_{\boldsymbol{X}}(\cdot,a) = P_{\boldsymbol{X},\boldsymbol{X}'}(:|\cdot,a) \otimes_{\boldsymbol{X}'} \Big(r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot,a,:) \oplus \big(\beta \odot v_{\boldsymbol{X}'}(:)\big)\Big).$$

Although this algorithm is very intuitive and also quite efficient in running time, it often suffers from its memory consumption since the complete transition probability tables $P_{\boldsymbol{X},\boldsymbol{X}'}(:|\cdot,a)$ can grow exponentially in the number of factors in the worst case.

The basic SPUDD method tries to solve the memory consumption problem by using factored CPTs directly without using the complete transition probability tables. The key computation in the value update rule is $\sum_{s' \in S} P(s'|s,a)f(s,s')$, where f is a function of s and

s'. For factored MDPs, this computation can be rewritten as

$$\begin{aligned}
&\sum_{s' \in S} P(s'|s,a) f(s,s') \\
&\quad = \sum_{\substack{x'_1 \in \Omega_{X_1} \\ x'_2 \in \Omega_{X_2} \\ \cdots \\ x'_n \in \Omega_{X_n}}} P(x'_1, x'_2, \ldots, x'_n | x_1, x_2, \ldots, x_n; a) f(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_n) \\
&\quad = \sum_{x'_1 \in \Omega_{X_1}} \sum_{x'_2 \in \Omega_{X_2}} \cdots \sum_{x'_n \in \Omega_{X_n}} \prod_{i=1}^{n} P\big(x'_i \big| \boldsymbol{\pi}'_i(a), a\big) f(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_n) \\
&\quad = \sum_{x'_1 \in \Omega_{X_1}} P\big(x'_1 \big| \boldsymbol{\pi}'_1(a), a\big) \sum_{x'_2 \in \Omega_{X_2}} P\big(x'_2 \big| \boldsymbol{\pi}'_2(a), a\big) \cdots \\
&\qquad\qquad \cdot \sum_{x'_n \in \Omega_{X_n}} P\big(x'_n \big| \boldsymbol{\pi}'_n(a), a\big) f(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_n). \qquad (3.38)
\end{aligned}$$

SPUDD performs this computation by eliminating next-time factors one by one. Let $\boldsymbol{X}'_{1,i} = \{X'_1, X'_2, \ldots, X'_i\}$. Then we have $\boldsymbol{X}'_{1,n} = \boldsymbol{X}'$. Let $f^n = f^n_{\boldsymbol{X},\boldsymbol{X}'}$ such that

$$\begin{aligned}
f^n(x_1, x_2, \ldots, x_n; &x'_1, x'_2, \ldots, x'_n) = r(s,a,s') + \beta v(s') \\
&= r(x_1, x_2, \ldots, x_n; a; x'_1, x'_2, \ldots, x'_n) + \beta v^{t-1}(x'_1, x'_2, \ldots, x'_n),
\end{aligned}$$

and let $f^{i-1} = f^{i-1}_{\boldsymbol{X},\boldsymbol{X}'_{1,i-1}}$ such that

$$\begin{aligned}
&f^{i-1}(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_{i-1}) \\
&\qquad = \sum_{x'_i \in \Omega_{X_i}} P\big(x'_i \big| \boldsymbol{\pi}'_i(a), a\big) \cdot f^i(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_i). \qquad (3.39)
\end{aligned}$$

Notice that f^i depends on factors $\boldsymbol{X} \cup \boldsymbol{X}'_{1,i}$, but f^{i-1} only depends on factors $\boldsymbol{X} \cup \boldsymbol{X}'_{1,i-1}$, that is, the factor X'_i has been eliminated. This is possible because we assume that the 2TBN is simple, therefore $\boldsymbol{\Pi}'_i(a)$ is a subset of $\boldsymbol{X}$ and contains no next-time factors. According to Eq. (3.38), we obtain $f^0 = f^0_{\boldsymbol{X}}$ where

$$f^0(x_1, x_2, \ldots, x_n) = \sum_{s' \in S} P(s'|s,a)\big(r(s,a,s') + \beta v^{t-1}(s')\big) = q^t(s,a).$$

The computation in Eq. (3.39) is performed using the generalized matrix multiplication:[19]

$$f^{i-1} = f^{i-1}_{\boldsymbol{X},\boldsymbol{X}'_{1,i-1}} = P\big(X'_i \big| \boldsymbol{\Pi}'_i(a), a\big) \otimes_{\{X'_i\}} f^i_{\boldsymbol{X},\boldsymbol{X}'_{1,i}}. \qquad (3.40)$$

[19] The original SPUDD code did not use the built-in operators from the CUDD package, but implemented this operation in a different way, which is less efficient.

Algorithm 3.10 SPUDD under the MER Objective

$v, \pi = \mathsf{SPUDD}(M, G, \mathcal{Y}, \beta, \epsilon)$

Input:
- $M = (\boldsymbol{X}, A, P, r)$, a factored MDP model;
- G, a set of goal states represented as a BDD;
- $\mathcal{Y}$, partitions of $\boldsymbol{X}$ into groups, one partition for each action;
- β, a discount factor, $0 < \beta < 1$;
- ϵ, an accuracy parameter, $\epsilon > 0$;

Output:
- π, an ϵ-optimal policy;
- v, an ϵ-optimal value function;

Local:
- f, intermediate ADDs;

```
for all $a \in A$ do
    for all $\boldsymbol{Y} \in \mathcal{Y}(a)$ do
        $P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \leftarrow 1$;
        for all $X \in \boldsymbol{Y}$ do
            $P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \leftarrow P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \odot P\big(X' \big| \boldsymbol{\Pi}'_{X}(a), a\big)$;
        end for
    end for
end for
$v_{\boldsymbol{X}}(\cdot) \leftarrow 0$;
repeat
    $v'_{\boldsymbol{X}}(\cdot) \leftarrow v_{\boldsymbol{X}}(\cdot)$;
    $v_{\boldsymbol{X}'}(:) \leftarrow v_{\boldsymbol{X}}(\cdot)[\boldsymbol{X}/\boldsymbol{X}']$;
    $v_{\boldsymbol{X}}(\cdot) \leftarrow -\infty$;
    for all $a \in A$ do
        $f \leftarrow r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot, a, :) \oplus (\beta \odot v_{\boldsymbol{X}'}(:))$;
        for all $\boldsymbol{Y} \in \mathcal{Y}(a)$ do
            $f \leftarrow P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \otimes_{\boldsymbol{Y}'} f$;
        end for
        $q_{\boldsymbol{X}}(\cdot, a) \leftarrow \text{ITE}(G, f, 0)$;
        $v_{\boldsymbol{X}}(\cdot) \leftarrow v_{\boldsymbol{X}}(\cdot) \ovee q_{\boldsymbol{X}}(\cdot, a)$;
    end for
until $\big\| v_{\boldsymbol{X}}(\cdot) \ominus v'_{\boldsymbol{X}}(\cdot) \big\| \leq \epsilon$;
obtain $\pi$ by choosing the maximizing actions;
```

In fact, we can eliminate the next-time factors in any order since they are uncorrelated. This allows us to group some factors together to make more efficient use of memory as the full version of SPUDD does.

The direct generalization of value iteration is more time efficient but consumes more memory, and the basic SPUDD is more memory efficient but consumes more time. The full version of SPUDD provides a tradeoff between the two extremes by grouping factors and using partially combined transition probability tables. For a given action a, we partition the set of factors into m groups $\mathcal{Y}(a) = \{\boldsymbol{Y}_1(a), \boldsymbol{Y}_2(a), \ldots, \boldsymbol{Y}_m(a)\}$. Then we can compute the probability $P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big)$ for $\boldsymbol{Y} \in \mathcal{Y}(a)$, where $\boldsymbol{\Pi}'_{\boldsymbol{Y}}(a) = \bigcup_{X \in \boldsymbol{Y}} \boldsymbol{\Pi}'_X(a)$. We next eliminate $\boldsymbol{Y}'_j(a)$'s one by one as in the basic SPUDD method.

The full version of SPUDD is shown as Algorithm 3.10 (SPUDD).[20] Lines 1–8 precompute the partially combined transition probability tables, and Lines 9–22 are the value iteration procedure. Line 12 performs the change of variable operation to prepare the next-time value function for Line 15. The q-functions are computed in Lines 15–19 following Eq. (3.40). The same termination condition as that for a naive value iteration applies (Line 22, which uses the values saved in Line 11). The policy can be retrieved from the value functions (Line 23) by checking for which action(s), the corresponding q-values agree with the value function.

3.7.2.3 Extensions to the Original SPUDD

In this section, we consider two extensions of the original SPUDD to deal with synchronic arcs and to improve the efficiency of planning.

The first extension extends the capability of SPUDD to deal with synchronic arcs. Such an extension has been considered for structural dynamic programming methods using decision trees (Boutilier, 1997; Boutilier *et al.*, 2000), but their algorithm is much more complicated since it manipulates the decision trees directly and the main ideas are clouded by details. Our extension uses a similar idea, but benefits from using standard ADD operations, and the resulting algorithm is much simpler. This extension is necessary for us to solve the painted-blocks example, since the predicates in the same outcome list are correlated (see Figure 3.24).

Synchronic arcs in 2TBNs introduce dependencies among next-time factors, and represent correlated action effects. In other words, the second layer of a 2TBN forms a non-trivial dag by itself. We refer to this dag as the next-time dag. In this case, we cannot eliminate factors in an arbitrary order. The reason is as follows. Suppose that the action a is given. We have an intermediate result f that depends on factors $\boldsymbol{X} \cup \boldsymbol{Y}' \cup \{Z'\}$ where $Z \notin \boldsymbol{Y}$ and $\boldsymbol{Y} \subset \boldsymbol{X}$, and we are going to eliminate factor Z' whose parents $\boldsymbol{\Pi}'_Z$ include other next-time factors, say $\boldsymbol{\Pi}'_Z = \boldsymbol{\Pi}_1 \cup \boldsymbol{\Pi}'_2$ where $\boldsymbol{\Pi}_1 \subseteq \boldsymbol{X}$ and $\boldsymbol{\Pi}'_2 \subseteq \boldsymbol{X}'$. The elimination can be done in

[20] In the original implementation, SPUDD uses a parameter *BIGADD* as the limit of the sizes of ADDs representing $P\big(\boldsymbol{Y}' \big| \boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big)$. Our formulation here is more general and more flexible.

a way similar to Eq. (3.39) as

$$\sum_{z'\in\Omega_Z} P\big(z'\big|\boldsymbol{\pi}'_Z, a\big) f(\boldsymbol{x}, \boldsymbol{y}', z') = \sum_{z'\in\Omega_Z} P\big(z'\big|\boldsymbol{\pi}_1, \boldsymbol{\pi}'_2; a\big) f(\boldsymbol{x}, \boldsymbol{y}', z'),$$

where $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}'_2$ are vectors of values for factors $\boldsymbol{\mathit{\Pi}}_1$ and $\boldsymbol{\mathit{\Pi}}'_2$, respectively. However, if $\boldsymbol{\mathit{\Pi}}'_2 \nsubseteq \boldsymbol{Y}$, then the result will not be a function of $\boldsymbol{X}$ and $\boldsymbol{Y}'$ only, but also a function of $\boldsymbol{\mathit{\Pi}}'_2$. In other words, new next-time factors can be introduced when eliminating a factor. Thus if handled improperly, some factors need to be eliminated more than once. Consider the 2TBN in Figure 3.20(b) as an example. If we eliminate B' before A', then eliminating A' will reintroduce B', which needs to be eliminated again.

Fortunately, we can eliminate the next-time factors in some specific orders that will not lead to multiple elimination. This is because the next-time dag is acyclic, and we can eliminate factors in the reverse direction of the arcs in the dag. If we eliminate factors one by one, this can be achieved by reordering the factors in an order that is the reverse of a topological sort of next-time factors in the next-time dag. Equivalently, we can always choose a leaf factor from the next-time dag, and remove the eliminated factor from the dag once the choice is made. For the above example, the factor A' must be eliminated before B', while other factors C' and D' can be eliminated before, after, or interleaved with A' and B'. If we also want to eliminate factors in groups as in the full version of SPUDD, the grouping must not introduce loops. This can be done by always grouping consecutive factors in a topological sort of the next-time dag, and elimination of groups also proceeds in this order. For the same example as above, we can eliminate the group $\{A', B'\}$ before or after the group $\{C', D'\}$, or eliminate the group $\{A', D'\}$ before the group $\{B', C'\}$, but cannot eliminate the group $\{B', C'\}$ before the group $\{A', D'\}$. Moreover, the algorithm that deals with synchronic arcs is the same as Algorithm 3.10 (SPUDD) except that the groups $\mathcal{Y}(a)$ are ordered properly for each action a.

The second extension makes the dynamic programming operations more efficient, by taking advantage of identity transition probability tables, which indicate that the value of a factor is unchanged. This case is exemplified by the E and F factors in Figure 3.20. These factors appear in the 2TBN only for dealing with the frame problem, namely, which factors are not affected by the action (McCarthy and Hayes, 1969). However, the original SPUDD does not identify such factors associated with identity transition probability tables, which however can be exploited to improve its efficiency. If a factor X is not affected by the action a, then $\boldsymbol{\Pi}'_X(a) = \{X\}$ and the associated CPT is $P(X'|X, a) = 1$ if and only if $X' = X$. Therefore, the factor elimination Eq. (3.39) is reduced to a simple composition operation

$$f' = P(X'|X, a) \otimes_{X'} f = f[X' \to X]$$

and, if f does not involve X, a change of variable operation

$$f' = P(X'|X, a) \otimes_{X'} f = f[X'/X].$$

Both operations are cheaper than a generalized matrix multiplication. The latter operation is more efficient than the former one, and multiple changes of variables can be performed together in one operation. It is also more efficient to perform the former operation on a smaller ADD than on a bigger ADD, which implies that we should process unaffected factors before dealing with other factors.

3.7.2.4 SPUDD for Maximizing Expected Exponential Utility of Total Rewards

For the $\mathsf{MEU}_{\text{exp}}$ objective, a variant of the value-update rule is as follows, which can be viewed as obtained through the pseudo-discount factor transformation (or the pseudo-probability transformation since they are equivalent for explicit probabilities):

$$v^t_{\text{exp}}(s) = \iota, \qquad s \in G, \tag{3.41}$$

$$v^t_{\text{exp}}(s) = \max_{a \in A_s} q^t_{\text{exp}}(s, a), \qquad s \in S \setminus G, \tag{3.42}$$

$$q^t_{\text{exp}}(s, a) = \sum_{s' \in S} P(s'|s, a)\gamma^{r(s,a,s')} v^{t-1}_{\text{exp}}(s'), \qquad s \in S \setminus G, a \in A. \tag{3.43}$$

If we define f^n as

$$\begin{aligned} f^n(x_1, x_2, \ldots, x_n; x'_1, x'_2, \ldots, x'_n) &= \gamma^{r(s,a,s')} v^{t-1}_{\text{exp}}(s') \\ &= \gamma^{r(x_1,x_2,\ldots,x_n;a;x'_1,x'_2,\ldots,x'_n)} v^{t-1}_{\text{exp}}(x'_1, x'_2, \ldots, x'_n), \end{aligned}$$

and compute f^i for all $i = 0, 1, \ldots, n-1$ as in Eq. (3.39), then it follows that

$$f^0(x_1, x_2, \ldots, x_n) = \sum_{s' \in S} \gamma^{r(s,a,s')} v^{t-1}_{\text{exp}}(s') = q^t_{\text{exp}}(s, a).$$

Algorithm 3.11 SPUDD under the $\mathsf{MEU}_{\text{exp}}$ Objective

$v, \pi = \mathsf{SPUDDExp}(M, G, \mathcal{Y}, \gamma, \epsilon)$

Input:

- $M = (\boldsymbol{X}, A, P, r)$, a factored MDP model;
- G, a set of goal states represented as a BDD;
- $\mathcal{Y}$, partitions of $\boldsymbol{X}$ into groups, one partition for each action and $\mathcal{Y}(a)$ contains m_a groups;
- γ, a risk parameter, $\gamma > 0, \gamma \neq 1$;
- ϵ, an accuracy parameter, $\epsilon > 0$;

Output:

- π, an ϵ-optimal policy;
- v, an ϵ-optimal value function;

Local:

- f, intermediate ADDs;

```
 1: for all $a \in A$ do
 2:     for all $\boldsymbol{Y} \in \mathcal{Y}(a)$ do
 3:         $P\big(\boldsymbol{Y}'\big|\boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \leftarrow 1$;
 4:         for all $X \in \boldsymbol{Y}$ do
 5:             $P\big(\boldsymbol{Y}'\big|\boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \leftarrow P\big(\boldsymbol{Y}'\big|\boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \odot P\big(X'\big|\boldsymbol{\Pi}'_{X}(a), a\big)$;
 6:         end for
 7:     end for
 8: end for
 9: $\boxed{v_{\boldsymbol{X}}(\cdot) \leftarrow \iota;}$
10: repeat
11:     $v'_{\boldsymbol{X}}(\cdot) \leftarrow v_{\boldsymbol{X}}(\cdot)$;
12:     $v_{\boldsymbol{X}'}(:) \leftarrow v_{\boldsymbol{X}}(\cdot)[\boldsymbol{X}/\boldsymbol{X}']$;
13:     $v_{\boldsymbol{X}}(\cdot) \leftarrow -\infty$;
14:     for all $a \in A$ do
15:         $\boxed{f \leftarrow \gamma^{r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot,a,:)} \odot v_{\boldsymbol{X}'}(:);}$
16:         for all $\boldsymbol{Y} \in \mathcal{Y}(a)$ do
17:             $f \leftarrow P\big(\boldsymbol{Y}'\big|\boldsymbol{\Pi}'_{\boldsymbol{Y}}(a), a\big) \otimes_{\boldsymbol{Y}'} f$;
18:         end for
19:         $\boxed{q_{\boldsymbol{X}}(\cdot, a) \leftarrow \text{ITE}(G, f, \iota);}$
20:         $v_{\boldsymbol{X}}(\cdot) \leftarrow v_{\boldsymbol{X}}(\cdot) \ovee q_{\boldsymbol{X}}(\cdot, a)$;
21:     end for
22: until $\big\|v_{\boldsymbol{X}}(\cdot) \ominus v'_{\boldsymbol{X}}(\cdot)\big\| \leq \epsilon$;
23: obtain $\pi$ by choosing the maximizing actions;
```

Therefore, the factor elimination procedure from the previous section, including using groups of factors, can be used for the $\mathsf{MEU}_{\text{exp}}$ objective without change, except for a different set of initial values. This is exactly what the pseudo-discount factor transformation does. It is clearer to show how this transformation works using ADD operators:

$$\begin{tikzcd}
f^n_{\boldsymbol{X},\boldsymbol{X}'} & = & r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot,a,:) \arrow[d] & \oplus & \big(& \beta \arrow[d] & \odot & v_{\boldsymbol{X}'}(:)\big) \arrow[d] \\
f^n_{\boldsymbol{X},\boldsymbol{X}'} & = & 0 & \oplus & \big(& \gamma^{r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot,a,:)} & \odot & v_{\boldsymbol{X}'}(:)\big)
\end{tikzcd}$$

The transformed algorithm is shown in Algorithm 3.11 ($\mathsf{SPUDDExp}$) and the transformed parts are highlighted. Notice that we also need to transform the initial values (Line 9) and the values of the goal states (Line 19).

The correctness of SPUDDExp is immediate since it is value iteration using ADDs. We can also use the online testing mechanism described in Section 3.2.3, as least in principle, to test the finiteness conditions Condition 3.2 and Condition 3.4. There are methods for calculating the strongly connected components (Xie and Beerel, 2000), and the spectral radius computation can be carried out using ADDs directly since it only requires arithmetic operations. To summarize, we have the following theorem.

Theorem 3.18. *Assume that Condition 2.1 (Finite Model) holds and that one of Condition 2.7 (Positive Model) and Condition 3.5 (Strictly Negative Model) holds. Then the values from SPUDDExp (with online testing if needed) converges to the optimal values if the optimal values are finite, and the method outputs "*`infinite values`*" otherwise.* □

3.7.2.5 SPUDD and Painted-Blocks Problems

SPUDD, however, cannot be used directly to solve painted-blocks problems represented in 2TBNs and ADDs due to invalid states. Recall that we extend the definition of actions so that they result in self-loops if they are not applicable. For invalid states, no action is applicable, and their total rewards are always negative infinity. However, solving the problem under either the MER or the $\mathsf{MEU}_{\mathrm{exp}}$ objective requires the optimal values to be finite, otherwise SPUDD does not terminate.

There are several approaches that can deal with this problem. The first approach is to redefine the actions for invalid states so that they have finite values. For invalid states, we can define the transition probabilities of actions arbitrarily since they cannot be reached anyway. Therefore it is possible to define actions in invalid states so that the next-time states are valid. For example, we can define actions for invalid states in painted-blocks problems so that they always result in the state where all blocks are on the table. However, this approach requires problem-specific definitions of actions, and it is therefore difficult to automate such definitions. The second approach is to figure out what states are valid and only perform dynamic programming on valid states. This can be done by a breadth-first search starting with the initial states, applying all actions and including the resulting states to the set of valid states, until the set of valid states does not change. In principle, we need to represent a boolean function that defines the boundary of valid states and invalid states. The problem with this approach is that the boolean function can be quite complex and thus is time consuming to obtain and to use in dynamic programming. The third approach is a search approach starting from the initial states and only including states when necessary. We take this approach and discuss the details in the following section.

3.7.3 Risk-Sensitive Symbolic LAO*

The SPUDD method can be combined with LAO* to solve even larger factored MDPs efficiently (Feng and Hansen, 2002). We refer to this method as symbolic LAO*. In this method, both the search and dynamic programming steps of LAO* are implemented using decision diagrams (ADDs or BDDs). In particular, the dynamic programming step is implemented using a version of SPUDD on a restricted domain. Since we have shown that both LAO* and SPUDD can be generalized to the $\mathsf{MEU}_{\text{exp}}$ objective using the pseudo-discount factor transformation, so is the symbolic LAO* method. We thus do not repeat the original MER version of the method, but present the $\mathsf{MEU}_{\text{exp}}$ version directly.

Symbolic LAO*Exp uses a variant of LAO*Exp with two differences to the one we have discussed in Section 3.5. The first one is the dynamic programming method. Since we will use SPUDD, which is a value iteration method, the termination condition for LAO* is changed to be also based on the approximation error to the optimal values. The second difference is the search. As we discussed before, the search part of LAO* does not update the values, therefore remains the same for both the MER and $\mathsf{MEU}_{\text{exp}}$ objectives. However, the search can benefit from using BDDs, which can represent sets of states, and BDD operations. To facilitate the search using BDDs, we expand all fringe states at once in the forward search. These two differences lead to the variant of LAO*Exp shown in Algorithm 3.12 (LAOStarSPUDD).

The forward search is performed in the same way as in the original LAO*, except that all operations are done using BDDs and BDD operations (Feng and Hansen, 2002). The search procedure is shown in Algorithm 3.13 (ForwardBackwardSearchBDD). We use S^a_π to indicate the set of states for which the current policy π dictates to execute action a. These sets are

Algorithm 3.12 LAO* for Factored MDPs Using SPUDD

π = LAOStarSPUDD(M, S_0, G, h, ϵ)

Input:

- $M = (\boldsymbol{X}, A, P, r)$, a factored MDP model;
- G, a set of goal states, $S_0 \cap G = \varnothing$;
- ϵ, an accuracy parameter, $\epsilon > 0$;
- S_0, a set of initial states;
- h, a heuristic function;

Output:

- π, an ϵ-optimal partial policy;

Local:

- v, the value function;
- E, the explicit subgraph;
- Z, the relevant states whose values are updated;
- δ, the error from SPUDDRestricted;
- B, the currently best solution graph;
- F, the fringe states;

```
$v_{\boldsymbol{X}}(\cdot) \leftarrow h_{\boldsymbol{X}}(\cdot)$;
$E \leftarrow \varnothing$; $B \leftarrow \varnothing$; $\pi \leftarrow \varnothing$; $F \leftarrow S_0$; $Z \leftarrow F$;
repeat
    $E \leftarrow E \cup F$;
    $v, \pi, \delta \leftarrow$ SPUDDExpRestricted$(M, Z, v)$;
    $B, F, Z \leftarrow$ ForwardBackwardSearchBDD$(M, S_0, \pi, E)$;
    $F \leftarrow F \setminus G$;
until $F = \varnothing$ and $\delta \leq \epsilon$;
```

used in Line 9 to determine what states the agent can result in under the current policy, using the Image procedure, which is implemented using existential abstraction. The Image procedure, also shown in Algorithm 3.13 (ForwardBackwardSearchBDD), calculates the image of a given set of states after executing an action, that is, the set of states that are reachable by executing an action from a state in the given set. In order to perform this calculation, we obtain for all actions $a \in A$ and all states $s, s' \in S$, the boolean transition function $T(s, a, s')$ from the transition probability table $P(s'|s, a)$ such that $T(s, a, s') = \mathrm{T}$ if and only if $P(s'|s, a) > 0$. The calculation itself is performed using the existential abstraction operator $\bowtie$ from the ADD package. We need to change variables so that the result represents a set of states at the current time.

The backward search is performed similarly.[21] It is combined with forward search since we need to initialize the policy with backward indices using S_π^a (Line 16). The backward search uses the PreImage procedure instead, which is also implemented using existential abstraction.

SPUDDExp on a restricted domain is shown in Algorithm 3.14 (SPUDDExpRestricted). This is possible by using a mask (the current best solution graph B) to restrict the set of

[21] Feng and Hansen (2002) mistakenly claimed that the backward search step can be omitted and $Z = B$.

Algorithm 3.13 Search in LAO* Using BDDs

$B, F, Z = \mathsf{ForwardBackwardSearchBDD}(M, S_0, \pi, E)$

Input:

- $M = (\boldsymbol{X}, A, P, r)$, a factored MDP model;
- π, an currently best partial policy;
- S_0, a set of initial states;
- E, the explicit subgraph;

Output:

- B, the set of states in the best solution graph;
- Z, the set of states on which to perform dynamic programming;
- F, the fringe states;

Local:

- *from*, source states;
- *to*, target states;

```
for all $a \in A$ do
    $S_\pi^a \leftarrow \pi^{-1}(a)$;
end for
$B \leftarrow S_0$;
$\mathit{from} \leftarrow B$;
while $\mathit{from} \neq \varnothing$ do
    $\mathit{to} \leftarrow \varnothing$;
    for all $a \in A$ do
        $\mathit{to} \leftarrow \mathit{to} \cup \mathsf{Image}\big(\mathit{from} \cap S_\pi^a, T(\cdot, a, :)\big)$;
    end for
    $F \leftarrow F \cup (\mathit{to} \setminus E)$;
    $\mathit{from} \leftarrow (\mathit{to} \cap E) \setminus B$;
    $B \leftarrow B \cup \mathit{from}$;
end while

for all $a \in A$ do
    $T_\pi^a \leftarrow \mathsf{Image}(S_\pi^a, T(\cdot, a, :))$;
end for
$Z \leftarrow F$;
$\mathit{to} \leftarrow Z$;
while $\mathit{to} \neq \varnothing$ do
    $\mathit{from} \leftarrow \varnothing$;
    for all $a \in A$ do
        $\mathit{from} \leftarrow \mathit{from} \cup \mathsf{PreImage}\big(\mathit{to} \cap T_\pi^a, T(\cdot, a, :)\big)$;
    end for
    $\mathit{to} \leftarrow \mathit{from} \setminus Z$;
    $Z \leftarrow Z \cup \mathit{to}$;
end while
```

$\mathit{target} = \mathsf{Image}(\mathit{source}, T)$

```
$\mathit{target} \leftarrow \big(\mathit{source} \bowtie_{\boldsymbol{X}} T_{\boldsymbol{X},\boldsymbol{X}'}(\cdot, :)\big)[\boldsymbol{X}'/\boldsymbol{X}]$;
```

$\mathit{source} = \mathsf{PreImage}(\mathit{target}, T)$

```
$\mathit{source} \leftarrow \big(\mathit{target}[\boldsymbol{X}/\boldsymbol{X}'] \bowtie_{\boldsymbol{X}'} T_{\boldsymbol{X},\boldsymbol{X}'}(\cdot, :)\big)$;
```

Algorithm 3.14 SPUDDExp with a Restricted Domain for LAO*Exp

$v_{\exp}, \pi, \delta = \mathsf{SPUDDExpRestricted}(M, Z, G, v_{\exp}, \gamma, \epsilon)$

Input:

- $M = (\boldsymbol{X}, A, P, r)$, a factored MDP model;
- G, a set of goal states;
- γ, a risk parameter, $\gamma > 0, \gamma \neq 1$;
- Z, the set of states where the policy will be improved;
- $v_{\exp}$, the currently best value function;
- ϵ, a desired accuracy, $\epsilon > 0$;

Output:

- $v_{\exp}$, the improved value function;
- δ, the error;
- π, an improved partial policy;

Local:

- $q_{\exp}$, the q-function;

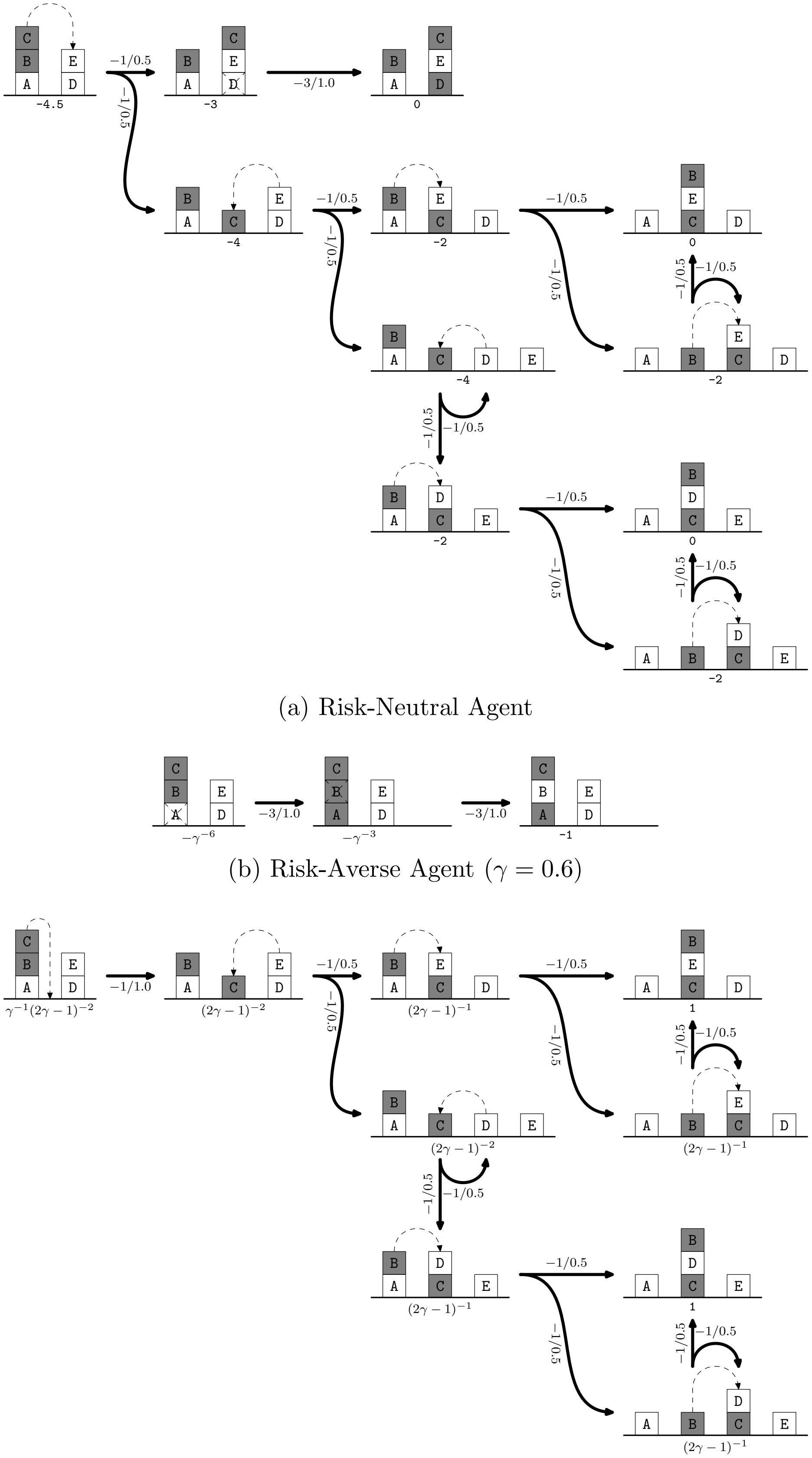
```
 1: $v_Z(\cdot) \leftarrow \textsc{ite}\big(Z, v_{\boldsymbol{X}}(\cdot), 0\big)$;
 2: $v_{\bar{Z}}(\cdot) \leftarrow \textsc{ite}\big(S \setminus Z, v_{\boldsymbol{X}}(\cdot), 0\big)$;
 3: $Z' \leftarrow \varnothing$;
 4: for all $a \in A$ do
 5:     $Z' \leftarrow Z' \cup \mathsf{Image}\big(Z, T(\cdot, a, :)\big)$;
 6: end for
 7: $Z'_1 \leftarrow Z' \cap Z$; $Z'_2 \leftarrow Z' \setminus Z$;
 8: $Z' \leftarrow Z'[\boldsymbol{X}/\boldsymbol{X}']$;
 9: $P_{Z,Z'}(: |\cdot, a) \leftarrow \textsc{ite}\big(Z \cup Z', P_{\boldsymbol{X},\boldsymbol{X}'}(: |\cdot, a), 0\big)$;
10: $r_{Z,Z'}(\cdot, a, :) \leftarrow \textsc{ite}\big(Z \cup Z', r_{\boldsymbol{X},\boldsymbol{X}'}(\cdot, a, :), 0\big)$;
11: repeat
12:     $v'_Z(\cdot) \leftarrow v_Z(\cdot)$;
13:     $v'_{Z'}(:) \leftarrow \textsc{ite}\Big(Z'_1, v'_Z(\cdot), \textsc{ite}\big(Z'_2, v_{\bar{Z}}(\cdot), 0\big)\Big)[\boldsymbol{X}/\boldsymbol{X}']$;
14:     $v_Z(\cdot) \leftarrow -\infty$;
15:     for all $a \in A$ do
16:         $q_Z(\cdot, a) \leftarrow P_{Z,Z'}(: |\cdot, a) \otimes_{\boldsymbol{X}'} \big(\gamma^{r_{Z,Z'}(\cdot,a,:)} \odot v'_{Z'}(:)\big)$;
17:         $v_Z(\cdot) \leftarrow v_Z(\cdot) \varovee q_Z(\cdot, a)$;
18:     end for
19:     $\delta \leftarrow \|v_Z(\cdot) \ominus v'_Z(\cdot)\|$;
20: until the termination condition is met;
21: obtain $\pi$ by choosing the maximizing actions for states in $Z$;
22: $v_{\boldsymbol{X}}(\cdot) \leftarrow \textsc{ite}\big(Z, v_Z(\cdot), v_{\bar{Z}}(\cdot)\big)$;
```

states that perform dynamic programming updates (Feng and Hansen, 2002). In general, the restriction of a function $f(x)$ on a set Z is

$$f_Z(x) = \begin{cases} f(x), & x \in Z \\ \\ \text{undefined}, & x \notin Z. \end{cases}$$

Therefore, the restriction of f on Z can be computed as $f_Z = \textsc{ite}(Z, f, \text{undefined})$ using the ADD operator ITE. In our implementation, it is convenient to use zero in place of the undefined value. Since a policy is represented as an ADD with integer values as indices to actions, -1 is used to denote an undefined action, which is needed for a partial policy.

For simplicity, we use the complete transition probability tables for all actions, represented as ADDs. Lines 1–2 separate the value function for those in Z and those not in Z



(a) Risk-Neutral Agent

(b) Risk-Averse Agent ($\gamma = 0.6$)

(c) Risk-Seeking Agent ($\gamma = 3.0$)

Figure 3.27: Optimal plans for the factored version of painted-blocks problem

since only values of states in Z will be changed. Next in Lines 3–6, we compute Z', the set of possible next-time states starting from a state in Z by executing an action. Line 7 partitions Z' into Z'_1, the part inside Z, and Z'_2, the part outside Z. Line 9 and Line 10 then calculate the restriction of the transition probability table and the reward function to the relevant states. The main loop is Lines 11–20, which is very similar to Algorithm 3.11 (SPUDDExp) except that all operations are performed on a restricted domain and Line 13 is needed to prepare for the next-time values needed in Line 16. Finally, Line 21 obtains the greedy policy for states in Z based on their values and Line 22 merges the two partial value functions.

The correctness of symbolic LAO*Exp follows directly from LAO*Exp and SPUDDExp.

Theorem 3.19. *Assume that Condition 2.1 (Finite Model) and Condition 3.5 (Strictly Negative Model) hold. Then values of the initial states from symbolic LAO* (with online testing if needed) converges to the optimal values if the optimal values of the initial states are finite, and the method outputs "*`infinite values`*" otherwise.* □

We apply symbolic LAO* and LAO*Exp to the painted-blocks problem, with the extensions discussed in Section 3.7.2.3. The resulting policies are shown in Figure 3.27. They are virtually the same as those obtained using LAO* or LAO*Exp with the flat representation (see Figure 3.6 and Figure 3.7). The only difference is that now the blocks are labeled with names. We break ties in favor of the action "working" on a block whose name is alphabetically smaller, which hence results in more states in the best solution graph than LAO* or LAO*Exp with the flat representation (see Section 3.5).

3.8 Summary

In this chapter, we discussed risk-sensitive planning with exponential utility functions, which model constant risk attitudes. The basic properties for planning under the $\mathsf{MEU}_{\text{exp}}$ objective have been developed in the operation research community. We therefore took a transformation-of-algorithms approach to solving large-scale planning problems by reusing ideas from decision-theoretic planning under a risk-neutral planning objective. Search, temporal abstraction, and state abstraction are three main symbolic strategies for solving large-scale problem in decision-theoretic planning under a risk-neutral objective. Using the transformation-of-algorithms approach, we showed that risk-neutral planners using any of these strategies can be transformed into risk-sensitive planners, and only small changes are made to the original algorithms. In particular, methods using temporal and state abstractions are transformed using the pseudo-probability and the pseudo-discount factor variants, respectively. We obtained risk-sensitive versions of several planners using the transformation and compared the resulting policies under risk-averse, risk-neutral, and risk-seeking objectives.

CHAPTER IV

GENERAL RISK-SENSITIVE UTILITY FUNCTIONS

In this chapter, we consider risk-sensitive planning objectives with general risk-sensitive utility functions. Exponential utility functions, as discussed in Chapter 3, are convenient to use, but cannot model all types of risk attitudes, for example, the risk attitude of a person who buys insurance and lottery tickets at the same time. Therefore, we need planning objectives and methods that deal with more general risk attitudes.

In general, a risk-sensitive utility function is a nonlinear real-valued function. We take a state-augmentation approach, and convert a problem under the MEU objective to a problem under the MER objective and reuse existing results from operations research for problems under the MER objective. However, the augmented state space is infinite in general, and we develop methods that deal with infinite state spaces, which is done using functional value functions and approximations.

In this chapter, we consider finite models and general risk-sensitive utility functions under the following monotonicity condition.

Condition 4.1 (Nondecreasing Utility Function)**.** The utility function $U(\cdot)$ is nondecreasing.

We show that finite horizon problems can be solved using a backward induction procedure (Section 4.2). For infinite horizon problems, we show that the optimal values can be approximated using value iteration if the utility function is asymptotically constant, linear, or exponential (Section 4.3). As an application of the results for MDPs with general utility functions, we also present an exact method for solving infinite horizon problems with one-switch utility functions (Section 4.4). We demonstrate how these methods work using the painted-blocks problem as an example.

4.1 Introduction

It is more difficult to solve risk-sensitive problems in general because the problem may not be decomposable any longer. For the purpose of dynamic programming, a problem is decomposable if we have a function $f_U(\cdot,\cdot)$ for a nonlinear utility function $U(\cdot)$ such that for all random variables $\mathbf{x}$ and $\mathbf{y}$,

$$E[U(\mathbf{x}+\mathbf{y})] = E\Big[f_U\big(\mathbf{x}, E[U(\mathbf{y})]\big)\Big].$$

In this way, a multi-stage reward process can be evaluated using a backward induction procedure: the expected utility of the total reward can be obtained by first evaluating the future expected utility of the total reward starting at the next decision epoch, and then combining the immediate reward with the future expected utility using the combing function f_U.

The identity utility function is decomposable, since we have the complete decomposition (see Section 2.3)

$$f(x,y) = x+y, \qquad \text{and} \qquad E[\mathbf{x}+\mathbf{y}] = E\big[\mathbf{x}+E[\mathbf{y}]\big] = E[\mathbf{x}]+E[\mathbf{y}];$$

and the exponential utility functions are also decomposable, since we have the partial decomposition (see Section 3.2)

$$f_{\exp}(x,y) = \gamma^x \cdot y, \qquad \text{and} \qquad E[U_{\exp}(\mathbf{x}+\mathbf{y})] = E\big[\gamma^{\mathbf{x}} \cdot E[U_{\exp}(\mathbf{y})]\big].$$

However, in general, we cannot find such a function f_U for an arbitrary utility function U satisfying Condition 4.1 (Nondecreasing Utility Function). Therefore, we must use a different approach to solving problems with general risk-sensitive utility functions. We take a state-augmentation approach and reduce a risk-sensitive problem to a risk-neutral problem with an augmented state space, to which existing results from the operations research literature can be applied.

4.2 *Finite Horizon*

This section considers finite horizon problems as the basis for dealing with infinite horizon problems, which are our true interest. This objective is denoted as MEU_T following the convention established in Table 2.1.

4.2.1 Basic Properties

First, we consider some basic properties under the MEU_T objective directly and establish the existence of an HD-optimal policy. This step is necessary to ensure that our state-augmentation approach can obtain an optimal policy.

The optimal values exist and are finite for finite horizon problems. For a given policy, the values exist and are finite for all states since there are only a finite number of trajectories. The optimal values exist since the values exists for all policies. The optimal values are finite since the total rewards are bounded and thus the expected utilities of the total reward for all policies are bounded as well.

To develop the optimality equations, we first consider the policy evaluation equations for a given policy. Define the expected utility of the total reward at decision epoch t given a history $h_t \in H_t$ for a policy $\pi \in \Pi$ as

$$v_{U,T}^{\pi,t}(h_t) = E^{h_t,\pi}\left[U\left(\sum_{\tau=0}^{T-1} r_\tau\right)\right] = E^{h_t,\pi}\left[U\left(w_T\right)\right].$$

Notice that at epoch t, the part of the trajectory $(a_t, s_{t+1}, \ldots, s_T)$ remains a random variable. It follows that $v_{U,T}^{\pi}(s) = v_{U,T}^{\pi,0}(s)$ for all states $s \in S$. We thus have the following theorem about the policy evaluation equations.

Theorem 4.1. *Let $\pi = (d_0, d_1, \ldots, d_{T-1}) \in \Pi$. The values $v_{U,T}^{\pi,t}(h_t)$ satisfy*

$$v_{U,T}^{\pi,T}(h_T) = U\left(\sum_{\tau=0}^{T-1} r_\tau\right) = U(w_T), \qquad h_T \in H_T,$$

$$v_{U,T}^{\pi,t}(h_t) = \sum_{a \in A_{s_t}} d_t(h_t, a) \sum_{s' \in S} P(s'|s_t, a) v_{U,T}^{\pi,t+1}\big(h_t \circ (a, s')\big), \qquad h_t \in H_t, \qquad 0 \le t < T,$$

where the last component of h_t is s_t.

Proof. Since no decision is made at $t = T$, the reward sequence is uniquely determined by h_T and the expectation can be dropped. Therefore, the results hold for $t = T$ by definition.

We show by induction that the results hold for $t < T$. Suppose the result holds for $t+1$. Then we have

$$\begin{aligned} v_{U,T}^{\pi,t}(h_t) &= E^{h_t,\pi}[U(w_T)] = E_{a_t,s_{t+1}}^{h_t,\pi}[U(w_T)] \\ &= E_{a_t,s_{t+1}}^{h_t,\pi}\left[E^{h_t,\pi}\left[U(w_T)\right| a_t = a, s_{t+1} = s']\right] \\ &= E_{a,s'}^{h_t,\pi}\left[E^{h_t\circ(a,s'),\pi}\left[U(w_T)\right]\right] = E_{a,s'}^{h_t,\pi}\left[v_{U,T}^{\pi,t+1}\big(h_t\circ(a,s')\big)\right] \\ &= \sum_{a\in A_{s_t}} d_t(h_t,a)\sum_{s'\in S} P(s'|s_t,a)v_{U,T}^{\pi,t+1}\big(h_t\circ(a,s')\big). \end{aligned}$$

Therefore, the result holds for all $0 \le t \le T$. □

Now we define

$$v_{U,T}^{*,t}(h_t) = \sup_{\pi\in\Pi} v_{U,T}^{\pi,t}(h_t),$$

and it follows that $v_{U,T}^{*}(s) = v_{U,T}^{*,0}(s)$ for all states $s \in S$. The optimality equations describe the relationship among the $v_{U,T}^{*,t}$ values.

Theorem 4.2. *The values $v_{U,T}^{*,t}(h_t)$ are the unique solution of the optimality equations*

$$\begin{aligned} v_{U,T}^{*,t}(h_T) &= U\left(\sum_{\tau=0}^{T-1} r_\tau\right) = U(w_T), && h_T \in H_T \\ v_{U,T}^{*,t}(h_t) &= \max_{a\in A_{s_t}} \sum_{s'\in S} P(s'|s_t,a)v_{U,T}^{*,t+1}\big(h_t\circ(a,s')\big), && h_t \in H_t, \quad 0 \le t < T, \end{aligned}$$

where the last component of h_t is s_t.

Proof. Suppose we have a solution $v_{U,T}^{t}(h_t)$ to the optimality equations. We show that $v_{U,T}^{t}(h_t) = v_{U,T}^{*,t}(h_t)$ for all histories $h_t \in H_t$ and all t with $0 \le t \le T$.

Since no decision is made at $t = T$, $v_{U,T}^{T}(h_T) = v_{U,T}^{\pi,T}(h_T)$ for all histories $h_T \in H_T$ and all policies $\pi \in \Pi$. Therefore, $v_{U,T}^{T}(h_T) = v_{U,T}^{*,T}(h_T)$ for all histories $h_T \in H_T$.

For $t < T$, we first show that $v_{U,T}^{t}(h_t) \ge v_{U,T}^{*,t}(h_t)$ and then that $v_{U,T}^{t}(h_t) \le v_{U,T}^{*,t}(h_t)$ for all histories $h_t \in H_t$.

We prove the first part by induction. We have shown the case for $t = T$. Suppose the result holds for $t+1$. Let $\pi = (d_0, d_1, \ldots, d_{T-1}) \in \Pi$ be an arbitrary policy. Then

$$\begin{aligned} v_{U,T}^{t}(h_t) &= \max_{a\in A_{s_t}} \sum_{s'\in S} P(s'|s_t,a)v_{U,T}^{t+1}\big(h_t\circ(a,s')\big) \\ &\ge \max_{a\in A_{s_t}} \sum_{s'\in S} P(s'|s_t,a)v_{U,T}^{*,t+1}\big(h_t\circ(a,s')\big) \end{aligned}$$

$$\geq \max_{a \in A_{s_t}} \sum_{s' \in S} P(s'|s_t, a) v_{U,T}^{\pi,t+1}\big(h_t \circ (a, s')\big)$$

$$\geq \sum_{a \in A_{s_t}} d_t(h_t, a) \sum_{s' \in S} P(s'|s_t, a) v_{U,T}^{\pi,t+1}\big(h_t \circ (a, s')\big) = v_{U,T}^{\pi,t}(h_t).$$

Since π is arbitrary, we have

$$v_{U,T}^{t}(h_t) \geq \sup_{\pi \in \Pi} v_{U,T}^{\pi,t}(h_t) = v_{U,T}^{*,t}(h_t).$$

For the second part of the proof, consider an HD policy $\pi' = (d_0', d_1', \cdots, d_{T-1}') \in \Pi^{\mathrm{HD}}$ such that

$$v_{U,T}^{t}(h_t) = \sum_{s' \in S} P(s'|s_t, d_t'(s_t)) v_{U,T}^{t+1}\big(h_t \circ (d_t'(s_t), s')\big),$$

which is possible since A is finite. Therefore

$$v_{U,T}^{t}(h_t) = v_{U,T}^{\pi',t}(h_t) \leq v_{U,T}^{*,t}(h_t).$$

Therefore, the values $v_{U,T}^{*,t}(h_t)$ are the unique solution to the optimality equations. □

The above results also show that there exists an HD-optimal policy. We can also compute an HD-optimal policy based on the optimality equations. However, the computation is impractical since one needs to enumerate all possible histories of length up to $2T + 1$. In the following sections, we instead use the state-augmentation approach to study additional structures of optimal policies and practical computational procedures.

4.2.2 Augmenting the State Space

We can restore the decomposability of the planning problem by augmenting the original state space with the accumulated rewards. Then we can still apply a backward induction procedure to determine an optimal policy.

Let

$$R = \{0\} \cup \big\{ r(s, a, s') \,\big|\, P(s'|s, a) > 0, s, s' \in S, a \in A_s \big\}$$

and

$$W^0 = \{0\}, \qquad W^{t+1} = \big\{ r + w \,\big|\, r \in R, w \in W^t \big\}, \qquad W = W^{\infty} = \bigcup_{t=0}^{\infty} W^t.$$

We refer to the elements of W^t as wealth levels. Since $0 \in R$, we have $W^t \subseteq W^{t+1}$ for all $t \in \mathbb{N}$. Obviously, W includes the total rewards of all possible trajectories of a T-horizon problem, including those trajectories terminating at a goal state (if any) in less than T steps.

We augment the state space with W, and denote all concepts concerning the augmented model by symbols enclosed in $\langle\rangle$. Notice that the augmentation depends on the utility function. More specifically, the reward function and the values of the augmented model depend on the utility function. Therefore, we use the notation $\langle r|_U\rangle$ and $\langle v|_U\rangle$ to specify the utility function for the rewards and values, respectively.

- The state space is $\langle S\rangle = S \times W$, and the set of goal states is $\langle G\rangle = G \times W$.
- The action space is $\langle A\rangle = A$, and for each augmented state $\langle s\rangle = (s, w) \in \langle S\rangle$, the set of available actions is $\langle A\rangle_{\langle s\rangle} = A_{\langle s\rangle} = A_s$. For this reason, we use a, $A_{\langle s\rangle}$, and A for the augmented model.
- The transition probabilities are defined for all $a \in A$ as:

  $$\langle P\rangle(\langle s\rangle'|\langle s\rangle, a) = \begin{cases} P(s'|s,a), & \langle s\rangle = (s,w), \langle s\rangle' = \big(s', w + r(s,a,s')\big), \\ 0, & \text{otherwise.} \end{cases}$$

- The reward function is defined as

  $$\langle r|_U\rangle(\langle s\rangle, a, \langle s\rangle') = U(w') - U(w)$$

  where $\langle s\rangle = (s, w)$ and $\langle s\rangle' = (s', w')$ for all valid transitions $(\langle s\rangle, a, \langle s\rangle')$, that is, for all augmented states $\langle s\rangle, \langle s\rangle' \in \langle S\rangle$, and all actions $a \in A_{\langle s\rangle}$ such that $\langle P\rangle(\langle s\rangle'|\langle s\rangle, a) = P(s'|s,a) > 0$.

Then we can have the usual definitions of histories, decision rules, and policies, all of which we denote with symbols enclosed in $\langle\rangle$, as $\langle h\rangle_t$, $\langle d\rangle_t$, and $\langle\pi\rangle$, respectively. For succinctness, we use the notation $\langle v|_U\rangle(\langle s\rangle) = \langle v|_U\rangle(s, w)$ instead of $\langle v|_U\rangle\big((s, w)\big)$.

In fact, for a finite horizon problem, it is sufficient to consider all wealth levels in W^t for each $0 \leq t \leq T$, but we use the set W to avoid time-dependent state spaces, which are cumbersome to deal with, since otherwise we need to define the augmented state space at epoch t to be $S \times W^t$.

Therefore, we are considering MDPs with countably infinite state spaces, but the set of available actions for each state remains finite, that is, Condition 2.2 (Countable States) holds for the augmented model. For clarity, we refer to the states of the original model as world states.

Next, we show that there exists a 1-1 mapping between a history of the original model and a class of equivalent histories of the augmented model, and use this result to show that the MEU_T objective for the original problem is equivalent to the MER_T objective for the augmented problem. The 1-1 mapping is established by the following two lemmata. The proofs of these lemmata benefit from our definition of history (see Section 2.2.1). The equivalence has been observed earlier, for example, in (White, 1987), but was not formalized. The incorrect results in (Bouakiz and Kebir, 1995; White, 1993) can be attributed to the lack of a formalization for a similar state-augmentation approach, as pointed out in (Wu and Lin, 1999).[1]

Lemma 4.3. *For any wealth level $w \in W$, and for any history of the original model, $h_t = (s_0, a_0, s_1, \dots, s_t) \in H_t$, the sequence $\langle h \rangle_t = (\langle s \rangle_0, a_0, \langle s \rangle_1, \dots, \langle s \rangle_t)$ is a history of the augmented model, where*

$$\langle s \rangle_k = (s_k, \tilde{w}_k) = \left(s_k, w + \sum_{\tau=0}^{k-1} r(s_\tau, a_\tau, s_{\tau+1}) \right) = (s_k, w + w_k), \qquad 0 \leq k \leq t. \tag{4.1}$$

Proof. By induction. Since $H_0 = S$ and $\langle H \rangle_0 = \langle S \rangle = S \times W$, the result is immediate for $t = 0$.

Suppose the result holds for some $t \geq 0$. For any history

$$h_{t+1} = h_t \circ (a_t, s_{t+1}) = (s_0, a_0, s_1, \dots, s_t, a_t, s_{t+1}) \in H_{t+1},$$

the induction hypothesis implies that for all $w \in W$, there exists a history $\langle h \rangle_t \in \langle H \rangle_t$ corresponding to h_t, and its last element is

$$\langle s \rangle_t = (s_t, \tilde{w}_t) = \left(s_t, w + \sum_{\tau=0}^{t-1} r(s_\tau, a_\tau, s_{\tau+1}) \right) \in \langle S \rangle.$$

Consider

$$\langle h \rangle_{t+1} = \langle h \rangle_t \circ \big(a_t, \langle s \rangle_{t+1}\big) = \langle h \rangle_t \circ (a_t, (s_{t+1}, \tilde{w}_{t+1})) = \langle h \rangle_t \circ \big(a_t, (s_{t+1}, \tilde{w}_t + r(s_t, a_t, s_{t+1}))\big).$$

[1] See Section 2.4.2 for more details.

Since $\tilde{w}_t \in W$, we have $\tilde{w}_{t+1} = \tilde{w}_t + r(s_t, a_t, s_{t+1}) \in W$. Since $h_{t+1} \in H_{t+1}$ by assumption, we have $a_t \in A_{s_t}$ and $P(s_{t+1}|s_t, a_t) > 0$. Therefore,

$$a_t \in A_{s_t} = \langle A \rangle_{(s_t, \tilde{w}_t)} = \langle A \rangle_{\langle s \rangle_t},$$

and

$$\langle P \rangle(\langle s \rangle_{t+1} | \langle s \rangle_t, a_t) = \langle P \rangle\big((s_{t+1}, \tilde{w}_{t+1}) \big| (s_t, \tilde{w}_t), a_t\big) = P(s_{t+1}|s_t, a_t) > 0,$$

so $\langle h \rangle_{t+1} \in \langle H \rangle_{t+1}$. □

Lemma 4.4. *For any history of the augmented model, $\langle h \rangle_t = (\langle s \rangle_0, a_0, \langle s \rangle_1, \ldots, \langle s \rangle_t) \in \langle H \rangle_t$ where $\langle s \rangle_k = (s_k, \tilde{w}_k)$ for all $0 \le k \le t$, there exists $w \in W$ such that*

$$\tilde{w}_k = w + \sum_{\tau=0}^{k-1} r(s_\tau, a_\tau, s_{\tau+1}) = w + w_k, \qquad 0 \le k \le t, \tag{4.2}$$

and the sequence $h_t = (s_0, a_0, s_1, \ldots, s_t)$ is a history of the original model.

Proof. By induction. The result is obvious for $t = 0$.

Suppose it holds for some $t \ge 0$. For $\langle h \rangle_{t+1} = \langle h \rangle_t \circ \big(a_t, \langle s \rangle_{t+1}\big) \in \langle H \rangle_{t+1}$, the induction hypothesis implies that the last element of $\langle h \rangle_t$, $\langle s \rangle_t$ can be written as $(s_t, w + w_t)$, and there exists a history $h_t \in H_t$ corresponding to $\langle h \rangle_t$, whose last element is s_t.

Consider $h_{t+1} = h_t \circ (a_t, s_{t+1})$. Since $\langle h \rangle_{t+1} \in \langle H \rangle_{t+1}$, we have

$$a_t \in \langle A \rangle_{\langle s \rangle_t} = A_{s_t} \qquad \text{and} \qquad \langle P \rangle(\langle s \rangle_{t+1} | \langle s \rangle_t, a_t) > 0,$$

which implies

$$P(s_{t+1}|s_t, a_t) = \langle P \rangle(\langle s \rangle_{t+1} | \langle s \rangle_t, a_t) > 0,$$

and

$$\tilde{w}_{t+1} = \tilde{w}_t + r(s_t, a_t, s_{t+1}) = w + w_t + r(s_t, a_t, s_{t+1}) = w + w_{t+1} = w + \sum_{\tau=0}^{t} r(s_\tau, a_\tau, s_{\tau+1}).$$

From $a_t \in A_{s_t}$ and $P(s_{t+1}|s_t, a_t) > 0$, it follows that $h_{t+1} \in H_{t+1}$. □

These two lemmata establish a 1-1 mapping between a history of the original model and a set of histories of the augmented model with the same sequence of world states and actions (but with different initial w). Define $\phi_w : H \mapsto \langle H \rangle$ to be the mapping from a history of the original model to the corresponding history of the augmented model with initial wealth level w according to Lemma 4.3, and define $\psi : \langle H \rangle \mapsto H$ to be the mapping from a history of the augmented model to the corresponding history of the original model according to Lemma 4.4. We summarize these observations in the following theorem.

Theorem 4.5. *For the functions ψ and ϕ_w defined as above, we have*

a. *For all histories $h \in H_t$ of the original model and all wealth levels $w \in W$, it holds that $h = \psi(\phi_w(h))$; and*

b. *For all augmented histories $\langle h \rangle \in \langle H \rangle_t$, there exists a wealth level $w \in W$ such that $\langle h \rangle = \phi_w\big(\psi(\langle h \rangle)\big)$.* □

We can then have a similar correspondence result for policies. For any policy in the original model, $\pi = (d_0, d_1, \ldots, d_{T-1}) \in \Pi^{\mathrm{HR}}$, we define a policy of the augmented model, $\Psi(\pi) = (\langle d \rangle_0, \langle d \rangle_1, \ldots, \langle d \rangle_{T-1}) \in \langle \Pi \rangle^{\mathrm{HR}}$, such that for all augmented histories $\langle h \rangle \in \langle H \rangle_t$,

$$\langle d \rangle_t(\langle h \rangle, a) = d_t\big(\psi(\langle h \rangle), a\big).$$

For any augmented policy $\langle \pi \rangle = (\langle d \rangle_0, \langle d \rangle_1, \ldots, \langle d \rangle_{T-1}) \in \langle \Pi \rangle^{\mathrm{HR}}$, we define a policy in the original model, $\Phi_w(\langle \pi \rangle) = (d_0, d_1, \ldots, d_{T-1}) \in \Pi^{\mathrm{HR}}$, such that for all histories $h \in H_t$,

$$d_t(h, a) = \langle d \rangle_t\big(\phi_w(h), a\big).$$

Notice that Ψ and Φ_w are mappings in the opposite direction to that of ψ and ϕ_w. It follows from the definitions that $\Phi_0(\Psi(\pi)) = \pi$. But in general, it does not hold that there exists w such that $\Psi(\Phi_w(\langle \pi \rangle)) = \langle \pi \rangle$.

The following theorem further relates the values and policies for the original model and the augmented model.

Theorem 4.6. *For each policy of the original model $\pi \in \Pi^{\mathrm{HR}}$, we have for all states $s \in S$,*

$$\langle v|_U \rangle_T^{\Psi(\pi)}(s, w) = E^{s,\pi}\left[U\left(w + \sum_{t=0}^{T-1} r_t\right)\right] - U(w) = E^{s,\pi}[U(w + w_T)] - U(w). \tag{4.3}$$

It immediately follows from the theorem that

$$v_{U,T}^{\pi}(s) = \langle v|_U \rangle_T^{\Psi(\pi)}(s, 0) + U(0), \qquad s \in S. \tag{4.4}$$

This formula relates the risk-sensitive values for the original model and the risk-neutral values for the augmented model.

We can consider $U(w+w')$ a utility function of w' obtained by shifting $U(w')$ to the left by w. It is convenient to denote this utility function as $U{\lll}w$. Therefore,

$$v^{\pi}_{U{\lll}w,T}(s) = E^{s,\pi}[U(w+w_T)],$$

and the result from Theorem 4.6 can be simplified to

$$v^{\pi}_{U{\lll}w,T}(s) = \langle v|_U\rangle_T^{\Psi(\pi)}(s,w) + U(w), \qquad s \in S.$$

Proof. First, we show that the two policies induce the same random processes of world states and actions with initial states s and (s,w) respectively. It is sufficient to show that for all wealth levels $w \in W$, all trajectories $h \in H_T$, all states $s \in S$, and all policies $\pi \in \Pi$, it holds that

$$\langle P\rangle^{(s,w),\Psi(\pi)}\big(\phi_w(h)\big) = P^{s,\pi}(h).$$

Let $\pi = (d_0, d_1, \ldots, d_{T-1})$ and $\Psi(\pi) = (\langle d\rangle_0, \langle d\rangle_1, \ldots, \langle d\rangle_{T-1})$. Also let $h = (s_0 = s, a_0, s_1, a_1, \ldots, s_T)$, $\phi_w(h) = (\langle s\rangle_0, a_0, \langle s\rangle_1, a_1, \ldots, \langle s\rangle_T)$, and $\langle h\rangle_t = \phi_w(h_t)$. According to Eq. (2.4), Theorem 4.5(a), and the definitions of the augmented model and $\Psi(\pi)$, it holds that for all t,

$$\langle d\rangle_t(\langle h\rangle_t, a) = d_t(\psi(\langle h\rangle_t), a) = d_t(\psi(\phi_w(h_t)), a) = d_t(h_t, a),$$

and thus

$$\begin{aligned}
&\langle P\rangle^{(s,w),\Psi(\pi)}\big(\phi_w(h)\big) = \langle P\rangle^{(s,w),\Psi(\pi)}(\langle s\rangle_0, a_0, \langle s\rangle_1, a_1, \ldots, \langle s\rangle_T)\\
&= \langle d\rangle_0(\langle s\rangle_0, a_0)\langle P\rangle(\langle s\rangle_1|\langle s\rangle_0, a_0)\langle d\rangle_1(\langle h\rangle_1, a_1)\langle P\rangle(\langle s\rangle_2|\langle s\rangle_1, a_1)\cdots\langle d\rangle_{T-1}(\langle h\rangle_{T-1}, a_{T-1})\langle P\rangle(\langle s\rangle_T|\langle s\rangle_{T-1}, a_{T-1})\\
&= d_0(s_0, a_0)P(s_1|s_0, a_0)d_1(h_1, a_1)P(s_2|s_1, a_1)\cdots d_{T-1}(h_{T-1}, a_{T-1})P(s_T|s_{T-1}, a_{T-1})\\
&= P^{s,\pi}(s_0, a_0, s_1, a_1, \ldots, s_T) = P^{s,\pi}(h).
\end{aligned}$$

Next, we show the equivalence of values. Suppose it holds that $\langle s\rangle_t = (s_t, \tilde{w}_t)$ for all $0 \le t \le T$. We have

$$\begin{aligned}
\langle v|_U\rangle_T^{\Psi(\pi)}(s,w) &= E^{(s,w),\Psi(\pi)}\left[\sum_{t=0}^{T-1}\langle r|_U\rangle_t\right] = E^{(s,w),\Psi(\pi)}\left[\sum_{t=0}^{T-1}\big(U(\tilde{w}_{t+1}) - U(\tilde{w}_t)\big)\right]\\
&= E^{(s,w),\Psi(\pi)}\left[\sum_{t=1}^{T}U(\tilde{w}_t) - \sum_{t=0}^{T-1}U(\tilde{w}_t)\right] = E^{(s,w),\Psi(\pi)}\big[U(\tilde{w}_T) - U(\tilde{w}_0)\big]\\
&= E^{(s,w),\Psi(\pi)}\left[U(w+w_T) - U(w)\right]
\end{aligned}$$

$$= E^{s,\pi}[U(w+w_T)] - U(w),$$

where the last equality is due to the fact that both policies produce the same random process of world states and actions. □

Theorem 4.7. *For each policy of the augmented model* $\langle\pi\rangle \in \langle\Pi\rangle^{\mathrm{HR}}$ *and each* $w \in W$*, it holds that*

$$v^{\Phi_w(\langle\pi\rangle)}_{U\lll w,T}(s) = \langle v|_U\rangle^{\langle\pi\rangle}_T(s,w) + U(w), \qquad s \in S. \tag{4.5}$$

It then follows that

$$v^{\Phi_0(\langle\pi\rangle)}_{U,T}(s) = \langle v|_U\rangle^{\langle\pi\rangle}_T(s,0) + U(0), \qquad s \in S.$$

Proof. Similar to the proof of Theorem 4.6. We first have that the two policies induce the same random processes of world states and actions with initial states s and (s,w) respectively. It is sufficient to show that for all wealth levels $w \in W$, all trajectories $\langle h\rangle \in \langle H\rangle_T$, all states $\langle s\rangle = (s,w) \in \langle S\rangle$, and all policies $\langle\pi\rangle \in \langle\Pi\rangle$, it holds that

$$\langle P\rangle^{(s,w),\langle\pi\rangle}(\langle h\rangle) = P^{s,\Phi_w(\langle\pi\rangle)}(\psi(\langle h\rangle)).$$

Let $\langle\pi\rangle = (\langle d\rangle_0, \langle d\rangle_1, \ldots, \langle d\rangle_{T-1})$ and $\Phi_w(\langle\pi\rangle) = (d_0, d_1, \ldots, d_{T-1})$. Also let $\langle s\rangle_0 = (s,w)$, $\langle h\rangle = (\langle s\rangle_0, a_0, \langle s\rangle_1, a_1, \ldots, \langle s\rangle_T)$, $\psi(\langle h\rangle) = (s_0, a_0, s_1, a_1, \ldots, s_T)$, and $\psi(\langle h\rangle_t) = h_t$. According to Eq. (2.4), Theorem 4.5(b), and the definition of $\Phi_w(\langle\pi\rangle)$, it holds that for all t,

$$d_t(h_t, a) = \langle d\rangle_t(\phi_w(h_t), a) = \langle d\rangle_t(\phi_w(\psi(\langle h\rangle_t)), a) = \langle d\rangle_t(\langle h\rangle_t, a),$$

and thus

$$P^{s,\Phi_w(\langle\pi\rangle)}(\psi(\langle h\rangle)) = P^{s,\Phi_w(\langle\pi\rangle)}(s_0, a_0, s_1, a_1, \ldots, s_T)$$
$$= d_0(s_0,a_0)P(s_1|s_0,a_0)d_1(h_1,a_1)P(s_2|s_1,a_1)\cdots d_{T-1}(h_{T-1},a_{T-1})P(s_T|s_{T-1},a_{T-1})$$
$$= \langle d\rangle_0(\langle s\rangle_0,a_0)\langle P\rangle(\langle s\rangle_1|\langle s\rangle_0,a_0)\langle d\rangle_1(\langle h\rangle_1,a_1)\langle P\rangle(\langle s\rangle_2|\langle s\rangle_1,a_1)\cdots\langle d\rangle_{T-1}(\langle h\rangle_{T-1},a_{T-1})\langle P\rangle(\langle s\rangle_T|\langle s\rangle_{T-1},a_{T-1})$$
$$= \langle P\rangle^{(s,w),\langle\pi\rangle}(\langle h\rangle).$$

Next, we show the equivalence of values. Suppose it holds that $\langle s\rangle_t = (s_t, \tilde{w}_t)$ for all $0 \le t \le T$. We have

$$\langle v|_U\rangle^{\langle\pi\rangle}_T(s,w) = E^{(s,w),\langle\pi\rangle}\left[\sum_{t=0}^{T-1}\langle r|_U\rangle_t\right] = E^{(s,w),\langle\pi\rangle}\left[\sum_{t=0}^{T-1}\left(U(\tilde{w}_{t+1}) - U(\tilde{w}_t)\right)\right]$$

$$
\begin{aligned}
&= E^{(s,w),\langle\pi\rangle}\left[\sum_{t=1}^{T} U(\tilde{w}_t) - \sum_{t=0}^{T-1} U(\tilde{w}_t)\right] = E^{(s,w),\langle\pi\rangle}\big[U(\tilde{w}_T) - U(\tilde{w}_0)\big] \\
&= E^{(s,w),\langle\pi\rangle}\left[U(w + w_T) - U(w)\right] \\
&= E^{s,\Phi_w(\langle\pi\rangle)}[U(w + w_T)] - U(w),
\end{aligned}
$$

where the last equality is due to the fact that both policies produce the same random process of world states and actions. □

4.2.3 Basic Properties Revisited

We now consider the basic properties for solving finite horizon risk-sensitive problems, based on the equivalence results and results for risk-neutral problems with a countable number of states. Now we can obtain the existence of optimal policies with more structure. The augmented models also allow us to adopt a functional interpretation of the value function.

Since the augmented model satisfies Condition 2.2 (Countable States), the results from Section 2.3.1 apply. The optimal values under the risk-neutral objective $\langle v\rangle^*_T(\langle s\rangle)$ exist and are finite for all augmented states. Moreover, there exists an MD-optimal policy $\langle\pi\rangle^*_T$ for the augmented model, and such a policy can be obtained using the backward induction procedure, at least in principle.

We now show that $\Phi_0(\langle\pi\rangle^*_T)$ is also an optimal policy for the problem under the MEU_T objective. Since we have shown in Section 4.2.1 that there exists an HD-optimal policy π^*_T under the MEU_T objective, we have

$$
\begin{aligned}
v^*_{U,T}(s) = v^{\pi^*_T}_{U,T}(s) &= \langle v|_U\rangle^{\Psi(\pi^*_T)}_T(s,0) + U(0) && \triangleright \text{ Theorem 4.6} \\
&\le \langle v|_U\rangle^*_T(s,0) + U(0) = \langle v|_U\rangle^{\langle\pi\rangle^*_T}_T(s,0) + U(0) = v^{\Phi_0(\langle\pi\rangle^*_T)}_{U,T}(s). && \triangleright \text{ Theorem 4.7}
\end{aligned}
$$

Therefore $\Phi_0(\langle\pi\rangle^*_T)$ is also an optimal policy under the MEU objective.

In terms of the original model, the optimal policy under MEU_T is no longer Markovian, but history-dependent. But the dependency on the history is only through the accumulated reward w. To distinguish such a policy from a usual HD policy, we call it an augmented-MD policy, or an aMD policy. If an aMD policy is optimal, we call it an aMD-optimal policy.

Based on the augmented model and the equivalence of optimal values, we can obtain a more practical backward induction procedure. A close examination of the augmented model

Algorithm 4.1 Backward Induction under the MEU_T Objective

$\langle v|_U\rangle^*_T = \mathsf{BackwardInductionUtility}(M, T, U)$

Input:
- $M = (S, A, P, r)$, a finite MDP model;
- T, a planning horizon;
- U, a utility function;

Output:
- $\langle v|_U\rangle^*_T$, the optimal values for the augmented model;

```
$t \leftarrow T$;
for all $s \in S$ do
    for all $w \in W^T$ do
        $\langle v|_U\rangle^{*,t}_T(s, w) \leftarrow 0$;
    end for
end for
for $t \leftarrow T - 1$ downto $0$ do
    for all $s \in S$ do
        for all $w \in W^t$ do
            $\langle v|_U\rangle^{*,t}_T(s, w) \leftarrow \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a)\Big[U(w + r(s, a, s')) - U(w) + \langle v|_U\rangle^{*,t+1}_T\big(s', w + r(s, a, s')\big)\Big]$;
        end for
    end for
end for
```

reveals that we need to care only about W^t at epoch t when doing backward induction, rather than the whole set of wealth levels W.[2] The reason is that we only care about augmented states of the form $(s, 0)$ at $t = 0$. Therefore, the backward induction procedure for a general utility function is shown in Algorithm 4.1 ($\mathsf{BackwardInductionUtility}$). Since there are only a finite number of elements in W^t for all t with $0 \leq t \leq T$, the procedure can be carried out. For practical reasons, we do not try to extract the policy explicitly, rather we keep the value functions and act greedily. This is especially convenient when we use the functional interpretation of the procedure (see below) and use approximations.

White (1987) presented another augmented model with a reward function which can take non-zero values only for the penultimate epoch. Based on his augmented model, he presented the plan evaluation and optimality equations without proofs. He also presented a slightly different backward induction procedure. Although both his and our methods result in the same values and optimal policies for finite horizon problems, our version can be extended to infinite horizon problems (see Section 4.3.4).

[2] Here we use W^t rather than W for computational purposes only. This usage is different from the earlier argument that we need W to avoid nonstationary state spaces, when we deal with equivalence results.

4.2.4 Functions as Values

In the augmented models, augmented states with the same world state and different wealth levels share the same transition probability distributions, so the backward induction updates for these augmented states are similar. This allows for another interpretation of the backward induction procedure for the augmented model. We can consider the value functions to be mappings from a world state to a real-valued function of wealth levels, instead of from an augmented state, that is, a world state and a wealth level, to a real number. Suppose a value function for the augmented model is $\langle v|_U \rangle : S \times W \mapsto \mathbb{R}$. We define

$$V_U(s, w) = \langle v|_U \rangle(s, w) + U(w).$$

Then we can consider V_U as a functional value function $V_U : S \mapsto (W \mapsto \mathbb{R})$, and $V_U(s, \cdot)$ denotes the functional value function of the state s.

Therefore, the backward induction procedure for the augmented model can be rewritten using functional value functions so that it can be seen as a standard backward induction procedure for the original model, where the values of states are now real-valued functions. The procedure can be simplified into a functional form as shown in Algorithm 4.2 (BackwardInductionUtilityFunctional). In this procedure, the dynamic programming operations can be carried out by operations on functions. For example, the value function $V_U(s, w + r)$ can be obtained by shifting the value function $V_U(s, w)$ to the left by r, which we denote as $V_U(s, \cdot) \lll r$ in a functional form. The value update rule (Line 7) can be verified as follows:

$$\begin{aligned}
&V_{U,T}^{*,t}(s, w) = \langle v|_U \rangle_T^{*,t}(s, w) + U(w) \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \Big[U(w + r(s, a, s')) - U(w) + \langle v|_U \rangle_T^{*,t+1}\big(s', w + r(s, a, s')\big) \Big] + U(w) \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \Big[U(w + r(s, a, s')) + \langle v|_U \rangle_T^{*,t+1}\big(s', w + r(s, a, s')\big) \Big] \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \cdot V_{U,T}^{*,t+1}(s', w + r(s, a, s')).
\end{aligned}$$

Algorithm 4.2 Backward Induction under the MEU_T Objective: Functional Form

```
$V_T = \mathsf{BackwardInductionUtilityFunctional}(M, T, U)$
Input:
    • $M = (S, A, P, r)$, a finite MDP model;
    • $T$, a planning horizon;
    • $U$, a utility function;
Output:
    • $V_T^*$, the functional optimal value function;

t ← T;
for all $s \in S$ do
    $V_{U,T}^{*,t}(s, \cdot) \leftarrow U(\cdot)$;
end for
for $t \leftarrow T - 1$ downto 0 do
    for all $s \in S$ do
        $V_{U,T}^{*,t}(s, \cdot) \leftarrow \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \cdot V_{U,T}^{*,t+1}(s', \cdot) \lll r(s, a, s')$;
    end for
end for
```

4.2.5 Approximations

This reformulation provides an opportunity to approximate the optimal values with a more efficient procedure. Although the backward induction procedure for the finite horizon expected utility objectives only considers the relevant wealth levels at the different decision epochs, the complexity of maintaining all wealth levels is still too high. Suppose that the number of distinct non-zero reward values is $n = |R \setminus \{0\}|$. In the worst case, the number of wealth levels to be considered at epoch t for the MEU_T objective is $\left|W^t\right| = \binom{t+n}{t}$, which is the number of combinations for selecting t items from $n + 1$ classes allowing repetition. This can be illustrated as follows. Suppose the distinct values in R are $0, r_1, r_2, \ldots, r_n$. $W^1 = R$ has elements

$$0, \quad r_1, \quad r_2, \quad \ldots, \quad r_n;$$

W^2 has elements

$$\begin{array}{ccccc}
0, & r_1, & r_2, & \ldots, & r_n, \\
 & 2r_1, & r_1 + r_2, & \ldots, & r_1 + r_n, \\
 & & 2r_2, & \ldots, & r_2 + r_n, \\
 & & & \ddots & \vdots \\
 & & & & 2r_n;
\end{array}$$

W^3 has elements

$$\begin{array}{ccccc}
0, & r_1, & r_2, & \dots, & r_n, \\
 & 2r_1, & r_1+r_2, & \dots, & r_1+r_n, \\
 & & 2r_2, & \dots, & r_2+r_n, \\
 & & & \ddots & \vdots \\
 & & & & 2r_n; \\
\hdashline
 & 3r_1, & 2r_1+r_2, & \dots, & 2r_1+r_n, \\
 & & r_1+2r_2, & \dots, & r_1+r_2+r_n, \\
 & & & \ddots & \vdots \\
 & & & & r_1+2r_n, \\
\hdashline
 & & 3r_2, & \dots, & 2r_2+r_n, \\
 & & & \ddots & \vdots \\
 & & & & r_2+2r_n, \\
\hdashline
 & & & \ddots & \vdots \\
\hdashline
 & & & & 3r_n;
\end{array}$$

and so on. Thus, the complexity of representing W^t explicitly to perform backward induction is $\binom{t+n}{t} = \frac{(t+n)\cdots(t+1)}{n!} = O(t^n)$, an n-th order polynomial in t, and thus computationally impractical if n is big. Therefore, some kind of approximation is desirable. It is possible that some of the elements in W^t can be obtained in more than one way. For example, if two rewards r_i and r_j are reducible, that is, there exists a pair of non-zero integers a and b, such that $ar_i = br_j$, then ar_i and br_j determine the same element in W^t if $t \geq \operatorname{lcm}(a, b)$, where $\operatorname{lcm}(a, b)$ is the least common multiplier of a and b. In this case, our counting $\binom{t+n}{t}$ is an overestimation. But it is not the case if the r_i's are irreducible, for example, the set of rewards $R = \{0, -1, -\sqrt{2}, -\sqrt{3}, -\sqrt{5}\}$. Therefore, this is a potential problem for efficiency and some kind of approximation of the value function is desirable.

For the approximation, we proceed by using the functional interpretation of the backward induction procedure for the finite horizon expected utility objectives, develop upper and lower bounds of the value functions, characterize the error of the greedy policy using the approximate value functions, and, as an example, use piecewise linear (PWL) functions as the tool for approximation.

The functional interpretation of values allows us to approximate the values efficiently, if the functional value functions can be approximated efficiently with, for example, piecewise linear functions. This is often the case since utility functions are usually rather smooth. We first approximate the true utility function, and then show that if we use a good approximation of the utility function in Algorithm 4.2 (BackwardInductionUtilityFunctional), the resulting value functions are also good approximations of the true value functions. We have the following results to show that the approximation error is not amplified through the backward induction procedure. The first two parts are special cases of the third part, but we list them explicitly since they are often more convenient to use.

Lemma 4.8. *Let U be a utility function.*[3]

a. *Suppose for a given error $\epsilon > 0$, we have an upper approximation $\overline{U}$ such that $U(\cdot) \leq \overline{U}(\cdot) \leq U(\cdot) + \epsilon$. Then, for all states $s \in S$ and all t with $0 \leq t \leq T$, we have $V^{*,t}_{U,T}(s,\cdot) \leq V^{*,t}_{\overline{U},T}(s,\cdot) \leq V^{*,t}_{U,T}(s,\cdot) + \epsilon$.*

b. *Suppose for a given error $\epsilon > 0$, we have a lower approximation $\underline{U}$ such that $U(\cdot) - \epsilon \leq \underline{U}(\cdot) \leq U(\cdot)$. Then, for all states $s \in S$ and all t with $0 \leq t \leq T$, we have $V^{*,t}_{U,T}(s,\cdot) - \epsilon \leq V^{*,t}_{\underline{U},T}(s,\cdot) \leq V^{*,t}_{U,T}(s,\cdot)$.*

c. *Suppose for given errors $\underline{\epsilon}, \overline{\epsilon} > 0$, we have an approximation $\cancel{U}$ such that $U(\cdot) - \underline{\epsilon} \leq \cancel{U}(\cdot) \leq U(\cdot) + \overline{\epsilon}$. Then, for all states $s \in S$ and all t with $0 \leq t \leq T$, we have $V^{*,t}_{U,T}(s,\cdot) - \underline{\epsilon} \leq V^{*,t}_{\cancel{U},T}(s,\cdot) \leq V^{*,t}_{U,T}(s,\cdot) + \overline{\epsilon}$.*

Proof. We only need to prove the result for part (c), since parts (a) and (b) are special cases of part (c). The proof is by (backward) induction on t. The result holds trivially for $t = T$. Suppose it holds for $t + 1$. We now show that it also holds for t. For all $s \in S$ and $w \in W$, we have

$$\begin{aligned} V^{*,t}_{\cancel{U},T}(s,w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V^{*,t+1}_{\cancel{U},T}(s', w + r(s,a,s')) \\ &\leq \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(V^{*,t+1}_{U,T}(s', w + r(s,a,s')) + \overline{\epsilon} \right) \end{aligned}$$

[3] *In this lemma and the following theorem, the results hold without Condition 4.1 (Nondecreasing Utility Function). In fact, we do not need Condition 4.1 for finite horizon problems. However, Condition 4.1 is needed for infinite horizon problems for the existence of values (see Section 4.3 and Chapter 5).*

$$= \left(\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V_{U,T}^{*,t+1}(s', w + r(s,a,s')) \right) + \overline{\epsilon} = V_{U,T}^{*,t}(s,w) + \overline{\epsilon}.$$

and

$$\begin{aligned}
V_{\sout{U},T}^{*,t}(s,w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V_{\sout{U},T}^{*,t+1}(s', w + r(s,a,s')) \\
&\geq \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(V_{U,T}^{*,t+1}(s', w + r(s,a,s')) - \underline{\epsilon} \right) \\
&= \left(\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V_{U,T}^{*,t+1}(s', w + r(s,a,s')) \right) - \underline{\epsilon} \\
&= V_{U,T}^{*,t}(s,w) - \underline{\epsilon}.
\end{aligned}$$

Therefore, the result holds. □

Next, we show that if we use the greedy policy based on these approximations, the error bounds of the value function are proportional to the planning horizon.

Theorem 4.9. *Let U be a utility function.*

- **a.** *Suppose for a given error $\epsilon > 0$, we have an upper approximation $\overline{U}$ such that $U(\cdot) \leq \overline{U}(\cdot) \leq U(\cdot) + \epsilon$, and let $\overline{\langle\pi\rangle}^*$ be an optimal augmented policy based on $\overline{U}$. Then $V_{U,T}^*(s) - \epsilon T \leq V_{U,T}^{\overline{\langle\pi\rangle}^*}(s) \leq V_{U,T}^*(s)$.*

- **b.** *Suppose for a given error $\epsilon > 0$, we have a lower approximation $\underline{U}$ such that $U(\cdot) - \epsilon \leq \underline{U}(\cdot) \leq U(\cdot)$, and let $\underline{\langle\pi\rangle}^*$ be an optimal augmented policy based on $\underline{U}$. Then $V_{U,T}^*(s) - \epsilon T \leq V_{U,T}^{\underline{\langle\pi\rangle}^*}(s) \leq V_{U,T}^*(s)$.*

- **c.** *Suppose for given errors $\underline{\epsilon}, \overline{\epsilon} > 0$, we have an approximation $\sout{U}$ such that $U(\cdot) - \underline{\epsilon} \leq \sout{U}(\cdot) \leq U(\cdot) + \overline{\epsilon}$, and let $\sout{\langle\pi\rangle}^*$ be an optimal augmented policy based on $\sout{U}$. Then $V_{U,T}^*(s) - \epsilon T \leq V_{U,T}^{\sout{\langle\pi\rangle}^*}(s) \leq V_{U,T}^*(s)$, where $\epsilon = \underline{\epsilon} + \overline{\epsilon}$.*

Proof. We prove the result for part (c). Parts (a) and (b) are special cases of part (c). Let $\sout{\langle\pi\rangle}^* = (\sout{\langle d\rangle}_0, \sout{\langle d\rangle}_1, \ldots, \sout{\langle d\rangle}_{T-1})$. We prove by induction that for $0 \leq t \leq T$, $V_{U,T}^{*,t}(s,w) - \epsilon(T-t) \leq V_{U,T}^{\sout{\langle\pi\rangle}^*,t}(s,w)$ for all $s \in S$ and all $w \in W$. Then the theorem holds by letting $t = 0$. The above result holds for $t = T$, since $V_{U,T}^{\sout{\langle\pi\rangle}^*,T}(s,w) = V_{U,T}^{*,T}(s,w) = U(w)$ for all $s \in S$ and all $w \in W^T$. Suppose it holds for $0 < t \leq T$. For any $s \in S$ and any $w \in W$, we have

$$V_{U,T}^{\sout{\langle\pi\rangle}^*,t-1}(s,w)$$

$$
\begin{aligned}
&= \sum_{s'} P(s'|s, \cancel{a}_{t-1}(s,w)) V_{U,T}^{\cancel{\pi}^*,t}(s', w + r(s, \cancel{a}_{t-1}(s,w), s')) \\
&\geq \sum_{s'} P(s'|s, \cancel{a}_{t-1}(s,w)) \left(V_{U,T}^{*,t}(s', w + r(s, \cancel{a}_{t-1}(s,w), s')) - \epsilon(T-t) \right) \\
&= \left(\sum_{s'} P(s'|s, \cancel{a}_{t-1}(s,w)) V_{U,T}^{*,t}(s', w + r(s, \cancel{a}_{t-1}(s,w), s')) \right) - \epsilon(T-t) \\
&\geq \left(\sum_{s'} P(s'|s, \cancel{a}_{t-1}(s,w)) \left(V_{\cancel{U},T}^{*,t}(s', w + r(s, \cancel{a}_{t-1}(s,w), s')) - \overline{\epsilon} \right) \right) - \epsilon(T-t) \quad \triangleright \quad \text{Lemma 4.8(c)} \\
&= \left(\sum_{s'} P(s'|s, \cancel{a}_{t-1}(s,w)) V_{\cancel{U},T}^{*,t}(s', w + r(s, \cancel{a}_{t-1}(s,w), s')) \right) - \overline{\epsilon} - \epsilon(T-t) \\
&= V_{\cancel{U},T}^{*,t-1}(s,w) - (\overline{\epsilon} + \epsilon(T-t)) \qquad \triangleright \quad \cancel{\pi}^* \text{ is optimal for the utility function } \cancel{U} \\
&\geq \left(V_{U,T}^{*,t-1}(s,w) - \underline{\epsilon} \right) - (\overline{\epsilon} + \epsilon(T-t)) \qquad \triangleright \quad \text{Lemma 4.8(c)} \\
&= V_{U,T}^{*,t-1}(s,w) - \epsilon(T-t+1) = V_{U,T}^{*,t-1}(s,w) - \epsilon(T-(t-1)).
\end{aligned}
$$

Therefore, the result holds. □

4.2.6 Piecewise Linear Approximations

The above results do not assume a particular form of approximation of the functional value functions. In this section, we show how the approximation can be done using (continuous) piecewise linear (PWL) approximations. Two reasons make PWL functions attractive. First, PWL functions have a finite representation, and the class of PWL functions is closed under the dynamic programming operations, which only includes scaling, shifting, addition, and maximization. Second, PWL functions can approximate any continuous function, so they can be used to approximate the optimal value functions with arbitrary continuous utility functions. In the following, we first define PWL functions formally, then show how utility functions can be approximated with PWL functions, and next show that dynamic programming updates can also be done efficiently using PWL functions.

Since we consider finite horizon problems, only a finite part of the utility function matters. A PWL function defined on a finite interval can be represented as a finite sequence of points sorted by their wealth levels,

$$(w^1, u^1), (w^2, u^2), \ldots, (w^\ell, u^\ell)$$

where ℓ is the number of points, and the following conditions apply:

$$w^1 < w^2 < \cdots < w^\ell, \qquad u^1 \leq u^2 \leq \cdots \leq u^\ell.$$

The points (w^i, u^i) are called breakpoints. The above representation defines a PWL function as follows

$$f_{\mathrm{PWL}}(w) = \frac{u^{i+1} - u^i}{w^{i+1} - w^i}(w - w^i) + u^i, \qquad w^i \leq w \leq w^{i+1}.$$

The initial upper and lower approximations of the original utility function can be done as follows. We approximate $U(w)$ on the finite interval $[w^1, w^\ell]$, which can be chosen based on W^T where T is the planning horizon for the problem to be solved. If the utility function is smooth and convex/concave on a given interval, we can use the sandwich method to approximate its upper and lower bounds within any given error, where the error decreases quadratically with the number of line segments, and the upper and lower approximations are monotonically nondecreasing as long as so is the original function (Rote, 1992). Usually, we can identify a finite number of smooth convex/concave segments of the utility function, so we can apply the sandwich method separately to each segment. The sandwich method for a smooth convex function is illustrated in Figure 4.1. The method proceeds in a recursive fashion. As shown in the figure, the thick curve is the function to be approximated, and we want to approximate it over the interval $[A_w, B_w]$. The first upper approximation is the chord AB obtained by connecting the two endpoints of the graph of the function. The first lower approximation consists of two line segments AC and BC, which are parts of the tangent lines at both endpoints. If the approximation error is too big, we recursively apply the procedure to the two subintervals $[A_w, C_w]$ and $[C_w, B_w]$, which are separated by the w-coordinate of the intersection point C from the lower approximation ACB. For example, the second upper approximation is given by AD and DB, and the second lower approximation is given by AE, EF and FB, where ED and DF are joined together since both are on the same line. Notice that the two endpoints A and B are on the resulting PWL functions.

If it is not easy to find the convex/concave segments of the utility function, an alternative approach is to use the method described in (Liu *et al.*, 1999). For a general continuous

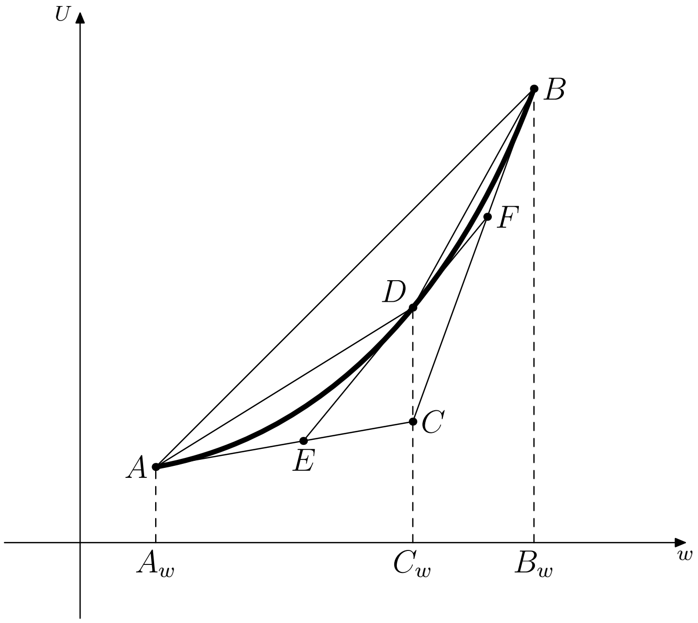


Figure 4.1: The sandwich method

function f, they argue that usually there exists a number $p > 1$, such that

$$g(x) = \left[f(x^{\frac{1}{p}})\right]^p$$

is a convex function. Next, the sandwich method can be applied to g to obtain $g_\pm$, the upper and lower approximations of g. Then the functions obtained by applying the following transformation to $g_\pm$ are upper and lower approximations of f:

$$f_\pm(x) = [g_\pm(x^p)]^{\frac{1}{p}}$$

Each of $f_\pm$ is a piecewise concave function, and can be approximated by applying the sandwich method once more.

We now demonstrate how to perform dynamic programming using PWL functions. Recall that the dynamic programming involves four types of basic operations: scaling, shifting, addition, and maximization. Scaling and shifting are operations involving a PWL function and a real number. They can be done by performing scaling or shifting on the breakpoints of the PWL function. Suppose that a PWL function $f(\cdot)$ is represented as a list of breakpoints

$$(w^1, u^1), (w^2, u^2), \ldots, (w^\ell, u^\ell).$$

Then the PWL function $k \cdot f(\cdot)$ is obtained as

$$(w^1, ku^1), (w^2, ku^2), \ldots, (w^\ell, ku^\ell),$$

and the PWL function $f(\cdot) \lll r$ is obtained as

$$(w^1 - r, u^1), (w^2 - r, u^2), \ldots, (w^\ell - r, u^\ell).$$

Addition and maximization are operations involving two PWL functions. We can perform these actions in two steps. Suppose we have another PWL function $g(\cdot)$ (with ℓ' breakpoints) represented as

$$(\hat{w}^1, \hat{u}^1), (\hat{w}^2, \hat{u}^2), \ldots, (\hat{w}^{\ell'}, \hat{u}^{\ell'}).$$

The first step is to merge the w values of the two PWL functions into a single list of w values (removing duplications) and compute their respective u values at these w values. Therefore, this step does not change the functions presented as PWL functions, but makes them represented with the same number of breakpoints. Suppose that this number is m. The second step then is to obtain the desired result. Suppose that after the first step, $f(\cdot)$ and $g(\cdot)$ are represented as

$$(\bar{w}^1, u_f^1), (\bar{w}^2, u_f^2), \ldots, (\bar{w}^m, u_f^m)$$

and

$$(\bar{w}^1, u_g^1), (\bar{w}^2, u_g^2), \ldots, (\bar{w}^m, u_g^m)$$

respectively. Then the function $f(\cdot) + g(\cdot)$ is obtained as

$$(\bar{w}^1, u_f^1 + u_g^1), (\bar{w}^2, u_f^2 + u_g^2), \ldots, (\bar{w}^m, u_f^m + u_g^m).$$

The function $\max\big(f(\cdot), g(\cdot)\big)$ is a little bit more complicated, since new breakpoints may be needed. Consider a segment over the interval $[\bar{w}^i, \bar{w}^{i+1}]$. Then on this interval, there are two possibilities. In the first case, one function dominates the other, and thus the maximum of the two functions is just the dominating function. In the second case, neither function dominates the other, and thus the two segments intersect. Then we need to add one more breakpoint and the result is the two segments that are connected at the breakpoint. For the efficiency of the maximization operation, an additional step is needed to remove unnecessary breakpoints and thus merge some subintervals into larger ones.

4.3 Infinite Horizon

In this section, we consider infinite horizon planning objectives derived from general risk-sensitive utility functions. We still use the state-augmentation approach. However, additional conditions are needed for this approach to be applicable. Therefore, we focus on positive and negative models in this section. Also for positive and negative models, we can develop computational methods to approximate the optimal values and optimal policies.

We first present the augmented model and discuss the necessity of existence conditions. Then we focus on positive and negative models in the later part of the section, and discuss a generalization of value iteration for the MEU objective.

4.3.1 The Augmented Model

For the MEU objective, we still use the state-augmentation approach. The augmented model is exactly the same as in the finite horizon case.

We expect a straightforward generalization of the results from the finite horizon case so that we still have the equivalence results. The equivalence results for histories are immediate, since the results from the finite horizon case, Lemma 4.3 and Lemma 4.4, still apply. We continue to use ϕ_w to denote the mapping from a history of the original model to the corresponding history of the augmented model with initial wealth level w, and ψ to denote the mapping from a history of the augmented model to the corresponding history of the original model. We also use $\varPsi$ to denote the related mapping from Π to $\langle\Pi\rangle$, and $\varPhi_w$ to denote the related mappings from $\langle\Pi\rangle$ to Π.

The following theorem relates the values and policies for the original model and the augmented model.

Theorem 4.10. *For each policy of the original model $\pi \in \Pi^{\mathrm{HR}}$, suppose the values $v^{\pi}_{U\lll w}(s)$ exist for all states $s \in S$ and all wealth levels $w \in W$. Then the augmented values $\langle v|_U\rangle^{\varPsi(\pi)}(s,w)$ exist for all states $s \in S$ and all wealth levels $w \in W$, and*

$$\langle v|_U\rangle^{\varPsi(\pi)}(s,w) = v^{\pi}_{U\lll w}(s) - U(w), \qquad s \in S. \tag{4.6}$$

It immediately follows from the theorem that if all the values exist, then

$$v_U^{\pi}(s) = \langle v|_U\rangle^{\Psi(\pi)}(s,0) + U(0), \qquad s \in S. \tag{4.7}$$

Proof. The results follow from Theorem 4.6 by letting T approach infinity. □

Theorem 4.11. *For each policy of the augmented model $\langle\pi\rangle \in \langle\Pi\rangle^{\mathrm{HR}}$, if the augmented values $\langle v|_U\rangle^{\langle\pi\rangle}(s,w)$ exist for all states $s \in S$ and the given wealth level w, then*

$$v_{U\lll w}^{\Phi_w(\langle\pi\rangle)}(s) = \langle v|_U\rangle^{\langle\pi\rangle}(s,w) + U(w), \qquad s \in S. \tag{4.8}$$

In particular, when the condition holds for $w = 0$, we have

$$v_U^{\Phi_0(\langle\pi\rangle)}(s) = \langle v|_U\rangle^{\langle\pi\rangle}(s,0) + U(0), \qquad s \in S.$$

Proof. The results follow from Theorem 4.7 by letting T approach infinity. □

It is important to ensure that for all augmented policies $\langle\pi\rangle \in \langle\Pi\rangle$, all states $s \in S$, and all wealth levels $w \in W$, the augmented values $\langle v|_U\rangle^{\langle\pi\rangle}(s,w)$ exist, so that we can apply results for countable state space models under the risk-neutral planning objective. This condition is equivalent to the the following one: for all policies $\pi \in \Pi$, all states $s \in S$, and all wealth levels $w \in W$, the values $v_{U\lll w}^{\pi}(s)$ exist. This condition is a stronger requirement than the one that for all policies $\pi \in \Pi$ and all states $s \in S$, the values $v_U^{\pi}(s)$ exist. In other words, it is possible that $v_U^{\pi}(s)$ exists but $v_{U\lll w}^{\pi}(s)$ does not exist for some $w \in W$.

A simple example is the model in Figure 2.2 when using the following utility function

$$U(w) = \begin{cases} w+1, & w < -1, \\ 0, & -1 \le w \le 1, \\ w-1, & w > 1. \end{cases}$$

For this example, $R = \{1, 0, -1\}$ and thus $W = \mathbb{Z}$. We have shown that starting from state s^1, the agent receives accumulated rewards 1 when the decision epoch is odd and 0 when the decision epoch is even. Since $U(1) = U(0) = 0$, the value of s^1 exists under the only policy π and $v_U^{\pi}(s^1) = 0$. Similarly, starting from state s^2, the agent receives accumulated rewards -1 and 0 when the decision epoch is odd and even, respectively.

Therefore, $v_U^\pi(s^2) = 0$. On the other hand, when $w = 1$, the value $v^\pi_{U\lll w}(s^1)$ does not exist since $U(w+1) = U(2) = 1 \neq U(w+0) = U(1) = 0$, but $v^\pi_{U\lll w}(s^2) = 0$. Similarly, when $w = -1$, the value $v^\pi_{U\lll w}(s^2)$ does not exist but $v^\pi_{U\lll w}(s^1) = 0$; when $w \neq 1, 0, -1$, neither value exists.

4.3.2 Existence and Finiteness of Optimal Values

In the rest of this chapter, we focus on negative and positive models, since we can develop computational methods for solving such problems under the MEU objective.

For negative and positive models, the expected utility of the total reward always exists, since the accumulated reward is monotonic with respect to the number of decision epochs. Moreover, the exponential utility functions are bounds for utility functions with which the risk-sensitive values are finite.

We first show that the values of the original model are finite, then we show that the values of the augmented model are also finite. Therefore, we can use methods for the augmented model to solve the original problem. We first consider negative models and then positive models.

Theorem 4.12. *Suppose Condition 2.1 (Finite Model), Condition 2.5 (Negative Model) and Condition 4.1 (Nondecreasing Utility Function) hold.*

a. *If Condition 3.2 (Negative Model with Finite Exponential Utilities) holds for some policy $\pi \in \Pi$ and some γ where $0 < \gamma < 1$, and if the utility function U satisfies $U(w) = O(\gamma^w)$ as $w \to -\infty$, then for this policy π and all states $s \in S$, the values $v_U^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_U^*(s)$ are finite.*

b. *If Condition 2.6 (Negative Model with Finite Expected Rewards) holds for some policy $\pi \in \Pi$, and if the utility function U satisfies $U(w) = O(w)$ as $w \to -\infty$, then for this policy π and all states $s \in S$, the values $v_U^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_U^*(s)$ are finite.*

Proof. **a.** If $U(w) = O(\gamma^w)$ as $w \to -\infty$ then there exist $C > 0$ and $D > 0$ such that for all $w \leq 0$, $U(w) \geq -C\gamma^w - D$. Therefore, for this special policy π, all states $s \in S$ and all finite T,

$$U(0) \geq v_{U,T}^{\pi}(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] = E^{s,\pi}[U(w_T)] \geq E^{s,\pi}\left[-C\gamma^{w_T} - D\right] = Cv_{\exp,T}^{\pi}(s) - D.$$

Thus,

$$U(0) \geq v_U^{\pi}(s) = \lim_{T\to\infty} v_{U,T}^{\pi}(s) \geq \lim_{T\to\infty}\left(Cv_{\exp,T}^{\pi}(s) - D\right) = Cv_{\exp}^{\pi}(s) - D > -\infty.$$

We also have

$$U(0) \geq v_U^*(s) = \sup_{\pi' \in \Pi} v_U^{\pi'}(s) \geq v_U^{\pi}(s) \geq Cv_{\exp}^{\pi}(s) - D > -\infty.$$

Therefore, the result holds.

b. If $U(w) = O(w)$ as $w \to -\infty$ then there exist $C > 0$ and $D > 0$ such that for all $w \leq 0$, $U(w) \geq Cw - D$. Therefore, for this special policy π, all states $s \in S$ and all finite T,

$$U(0) \geq v_{U,T}^{\pi}(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] = E^{s,\pi}[U(w_T)] \geq E^{s,\pi}\left[Cw_T - D\right] = Cv_T^{\pi}(s) - D.$$

Thus,

$$U(0) \geq v_U^{\pi}(s) = \lim_{T\to\infty} v_{U,T}^{\pi}(s) \geq \lim_{T\to\infty}\left(Cv_T^{\pi}(s) - D\right) = Cv^{\pi}(s) - D > -\infty.$$

We also have

$$U(0) \geq v_U^*(s) = \sup_{\pi' \in \Pi} v_U^{\pi'}(s) \geq v_U^{\pi}(s) \geq Cv^{\pi}(s) - D > -\infty.$$

Therefore, the result holds. □

Corollary 4.13. *Suppose Condition 2.1 (Finite Model), Condition 2.5 (Negative Model) and Condition 4.1 (Nondecreasing Utility Function) hold.*

a. *If Condition 3.2 (Negative Model with Finite Exponential Utilities) holds for some policy $\pi \in \Pi$ and some γ where $0 < \gamma < 1$, and if the utility function U satisfies $U(w) = O(\gamma^w)$ as $w \to -\infty$, then for this policy π, all states $s \in S$, and all wealth levels $w' \in W$, the values $v_{U\lll w'}^{\pi}(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_{U\lll w'}^*(s)$ are finite.*

b. *If Condition 2.6 (Negative Model with Finite Expected Rewards) holds for some policy $\pi \in \Pi$, and if the utility function U satisfies $U(w) = O(w)$ as $w \to -\infty$, then for this policy π, all states $s \in S$, and all wealth levels $w' \in W$, the values $v_{U\lll w'}^{\pi}(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_{U\lll w'}^*(s)$ are finite.*

Proof. For the first part, it is sufficient to show that for all $w' \in W$, $(U \lll w')(w) = O(\gamma^w)$, which holds since

$$(U \lll w')(w) = U(w + w') = O(\gamma^{w+w'}) = O(\gamma^{w'} \cdot \gamma^w) = O(\gamma^w),$$

and then Theorem 4.12 applies. The second part is similar. □

For positive models, we can obtain similar results.

Theorem 4.14. *Suppose Condition 2.1 (Finite Model), Condition 2.7 (Positive Model), and Condition 4.1 (Nondecreasing Utility Function) hold.*

a. *If Condition 3.4 (Positive Model with Finite Exponential Utilities) holds for some γ where $\gamma > 1$, and if the utility function U satisfies $U(w) = O(\gamma^w)$ as $w \to \infty$, then for all policies $\pi \in \Pi$ and all states $s \in S$, the values $v_U^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_U^*(s)$ are finite.*

b. *If Condition 2.8 (Positive Model with Finite Expected Rewards) holds, and if the utility function U satisfies $U(w) = O(w)$ as $w \to \infty$, then for all policies $\pi \in \Pi$ and all states $s \in S$, the values $v_U^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_U^*(s)$ are finite.*

Proof. **a.** If $U(w) = O(\gamma^w)$ as $w \to \infty$ then there exist $C > 0$ and $D > 0$ such that for all $w \geq 0$, $U(w) \leq C\gamma^w + D$. Therefore, for all policies $\pi \in \Pi$, all states $s \in S$ and all finite T,

$$U(0) \leq v_{U,T}^\pi(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] = E^{s,\pi}[U(w_T)] \leq E^{s,\pi}\left[C\gamma^{w_T} + D\right] = Cv_{\exp,T}^\pi(s) + D.$$

Thus,

$$U(0) \leq v_U^\pi(s) = \lim_{T\to\infty} v_{U,T}^\pi(s) \leq \lim_{T\to\infty}\left(Cv_{\exp,T}^\pi(s) + D\right) = Cv_{\exp}^\pi(s) + D < \infty.$$

Since Condition 3.4 implies for all states $s \in S$, the values $v_{\exp}^*(s)$ are finite, we have

$$U(0) \leq v_U^*(s) = \sup_{\pi\in\Pi} v_U^\pi(s) \leq \sup_{\pi\in\Pi}\left(Cv_{\exp}^\pi(s) + D\right) = Cv_{\exp}^*(s) + D < +\infty.$$

Therefore, the result holds.

b. If $U(w) = O(w)$ as $w \to \infty$ then there exist $C > 0$ and $D > 0$ such that for all $w \geq 0$, $U(w) \leq Cw + D$. Therefore, for all policies $\pi \in \Pi$, all states $s \in S$ and all finite T,

$$U(0) \leq v_{U,T}^\pi(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] = E^{s,\pi}[U(w_T)] \leq E^{s,\pi}\left[Cw_T + D\right] = Cv_T^\pi(s) + D.$$

Thus,

$$U(0) \le v_U^\pi(s) = \lim_{T\to\infty} v_{U,T}^\pi(s) \le \lim_{T\to\infty} (Cv_T^\pi(s) + D) = Cv^\pi(s) + D < \infty.$$

Since Condition 2.8 (Positive Model with Finite Expected Rewards) implies for all states $s \in S$, the values $v^*(s)$ are finite, we have

$$U(0) \le v_U^*(s) = \sup_{\pi\in\Pi} v_U^\pi(s) \le \sup_{\pi\in\Pi} (Cv^\pi(s) + D) = Cv^*(s) + D < +\infty.$$

Therefore, the result holds. □

Corollary 4.15. *Suppose Condition 2.1 (Finite Model), Condition 2.7 (Positive Model), and Condition 4.1 (Nondecreasing Utility Function) hold.*

a. *If Condition 3.4 (Positive Model with Finite Exponential Utilities) holds for some γ where $\gamma > 1$, and if the utility function U satisfies $U(w) = O(\gamma^w)$ as $w \to \infty$, then for all policies $\pi \in \Pi$, all states $s \in S$, and all wealth levels $w' \in W$, the values $v_{U\lll w'}^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_{U\lll w'}^*(s)$ are finite.*

b. *If Condition 2.8 (Positive Model with Finite Expected Rewards) holds, and if the utility function U satisfies $U(w) = O(w)$ as $w \to \infty$, then for all policies $\pi \in \Pi$, all states $s \in S$, and all wealth levels $w' \in W$, the values $v_{U\lll w'}^\pi(s)$ are finite. Therefore, for all states $s \in S$, the optimal values $v_{U\lll w'}^*(s)$ are finite.*

Proof. The same as that for Corollary 4.13. □

Therefore, under the conditions of Corollary 4.13 and Corollary 4.15, all values with the utility function $U\lll w$ with $w \in W$ in the original model are finite, and thus all values in the augmented model are finite. Therefore, the equivalence results from Theorem 4.10 and Theorem 4.11 holds.

4.3.3 Existence of Optimal Policies

For negative models, it follows from the theory for negative models under the MER objective that there exists an SD-optimal policy for the augmented model under the MER objective, since $\langle S \rangle$ is countable and A is finite (Theorem 7.3.6 in Puterman, 1994, also see Section 2.3.3). Therefore, there exists an aSD-optimal policy for the original model under the MEU objective.

For positive models, however, it is uncertain whether there exists an SD-optimal policy. The results for positive models under the MER objective imply that for any $\epsilon > 0$, there exists an MD policy $\langle\pi\rangle$ for the augmented model under the MER objective such that for all $\langle s\rangle \in \langle S\rangle$, $\langle v|_U\rangle^{\langle\pi\rangle}(\langle s\rangle) \geq (1-\epsilon)\langle v|_U\rangle^*(\langle s\rangle)$ (Theorem 7.2.7 in Puterman, 1994).

4.3.4 Value Iteration and Approximation

The value iteration procedure can be used to approximate the optimal values for the augmented models under the MER objective, that is, the optimal values for the original model under the MEU objective.

We can use a functional form of value iteration, since value iteration can be viewed as a generalization of backward induction for infinite horizon problems. The original value update rule for the augmented model is

$$\begin{aligned}\langle v|_U\rangle^{t+1}(\langle s\rangle) &= \max_{a\in A_s} \sum_{\langle s\rangle'\in\langle S\rangle} \langle P\rangle(\langle s\rangle'|\langle s\rangle, a)\big[\langle r|_U\rangle(\langle s\rangle, a, \langle s\rangle') + \langle v|_U\rangle^t(\langle s\rangle')\big]\\ &= \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a)\big[U(w+r(s,a,s')) - U(w) + \langle v|_U\rangle^t(s', w+r(s,a,s'))\big].\end{aligned}$$

Similar to the finite horizon case, we can define $V_U(s,w) = \langle v|_U\rangle(s,w) + U(w)$, and the value update rule then is

$$V_U^{t+1}(s,\cdot) = \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a) V_U^t(s',\cdot) \lll r(s,a,s'),$$

and the initial values should be $V^0(s,\cdot) = U(\cdot)$. The value iteration procedure is shown in Algorithm 4.3 (ValueIterationUtilityFunctional). From the results for negative and positive models with countable state spaces, it follows that value iteration converges to the optimal values (Puterman, 1994, see also Section 2.3.3.2).

As we have shown in Section 4.2, we need approximations to deal with the augmented state space since it is infinite. The following result shows that the approximation error of the value functions does not increase as value iteration proceeds. This result is similar to that in the finite horizon case, and the proof is the same as that of Lemma 4.8 after a substitution of variables.

Algorithm 4.3 Value Iteration under the MEU Objective: Functional Form

V = ValueIterationUtilityFunctional(M, U)

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- U, a utility function;

Output:

- V, a functional value function;

```
$t \leftarrow 0$;
for all $s \in S$ do
    $V_U^t(s, \cdot) \leftarrow U(\cdot)$;
end for
repeat
    for all $s \in S$ do
        $V_U^{t+1}(s, \cdot) \leftarrow \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \cdot V_U^t(s', \cdot) \lll r(s, a, s')$;
    end for
until $\left\| V_U^{t+1} - V_U^t \right\|$ is sufficiently small;
```

Theorem 4.16. *Let U be a utility function, and V_U^t be the value function at step t of value iteration.*

a. *Suppose for a given error $\epsilon > 0$, we have an upper approximation $\overline{U}$ such that $U(\cdot) \leq \overline{U}(\cdot) \leq U(\cdot) + \epsilon$. Then, for all states $s \in S$ and all $t \in \mathbb{N}$, we have $V_U^t(s, \cdot) \leq V_{\overline{U}}^t(s, \cdot) \leq V_U^t(s, \cdot) + \epsilon$.*

b. *Suppose for a given error $\epsilon > 0$, we have a lower approximation $\underline{U}$ such that $U(\cdot) - \epsilon \leq \underline{U}(\cdot) \leq U(\cdot)$. Then, for all states $s \in S$ and all $t \in \mathbb{N}$, we have $V_U^t(s, \cdot) - \epsilon \leq V_{\underline{U}}^t(s, \cdot) \leq V_U^t(s, \cdot)$.*

c. *Suppose for given errors $\underline{\epsilon}, \overline{\epsilon} > 0$, we have an approximation $\cancel{U}$ such that $U(\cdot) - \underline{\epsilon} \leq \cancel{U}(\cdot) \leq U(\cdot) + \overline{\epsilon}$. Then, for all states $s \in S$ and all $t \in \mathbb{N}$, we have $V_U^t(s, \cdot) - \underline{\epsilon} \leq V_{\cancel{U}}^t(s, \cdot) \leq V_U^t(s, \cdot) + \overline{\epsilon}$.* □

However, in order to use the PWL approximations, we need a method that can deal with functional value functions on an infinite interval. We show that this can be done for utility functions that are asymptotically constant, linear, or exponential. We next use negative models to demonstrate how to perform value iteration using a finite representation if such utility functions are used. The methods for positive models are similar. For negative models, we only need to consider the part of the utility function on the negative real half-line. We consider both the upper and lower approximations, since there are noticeable differences.

For each approximation, there are different ways of dealing with the infinite interval, and what we present here is just one possibility.

4.3.4.1 Asymptotically Constant Utility Functions

A utility function $U(w)$ is asymptotically constant as $w \to -\infty$ if there exists a constant U_-, such that

$$\lim_{w \to -\infty} U(w) = U_-.$$

According to Condition 4.1 (Nondecreasing Utility Function), the utility function $U(\cdot)$ is monotonically nondecreasing in the wealth level. Therefore, we have $U_- \leq U(w)$ for all w.

First, we consider a lower approximation $\underline{U}(\cdot)$ of $U(\cdot)$, as illustrated in Figure 4.2. Then for a given error $\epsilon > 0$, we can determine $\underline{w} \leq 0$ such that for all $w \leq \underline{w}$, it holds that

$$U(w) - \frac{\epsilon}{2} \leq U_- \leq U(w).$$

We now show that we only need to represent value functions on the interval $[\underline{w}, 0]$, since the model is negative. On the infinite interval $(-\infty, \underline{w}]$, we can approximate $U(\cdot)$ from below with an error no greater than $\frac{\epsilon}{2}$ using the constant U_-, as illustrated by the dashed segment in Figure 4.2(a). On the finite interval $[\underline{w}, 0]$, we can approximate $U(\cdot)$ from below with error $\frac{\epsilon}{2}$ using the methods described in Section 4.2.6, denoted as $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ where $\underline{U}_{\frac{\epsilon}{2}}(\underline{w}) = U(\underline{w})$ and $\underline{U}_{\frac{\epsilon}{2}}(0) = U(0)$ such that for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \frac{\epsilon}{2} \leq \underline{U}_{\frac{\epsilon}{2}}(w) \leq U(w).$$

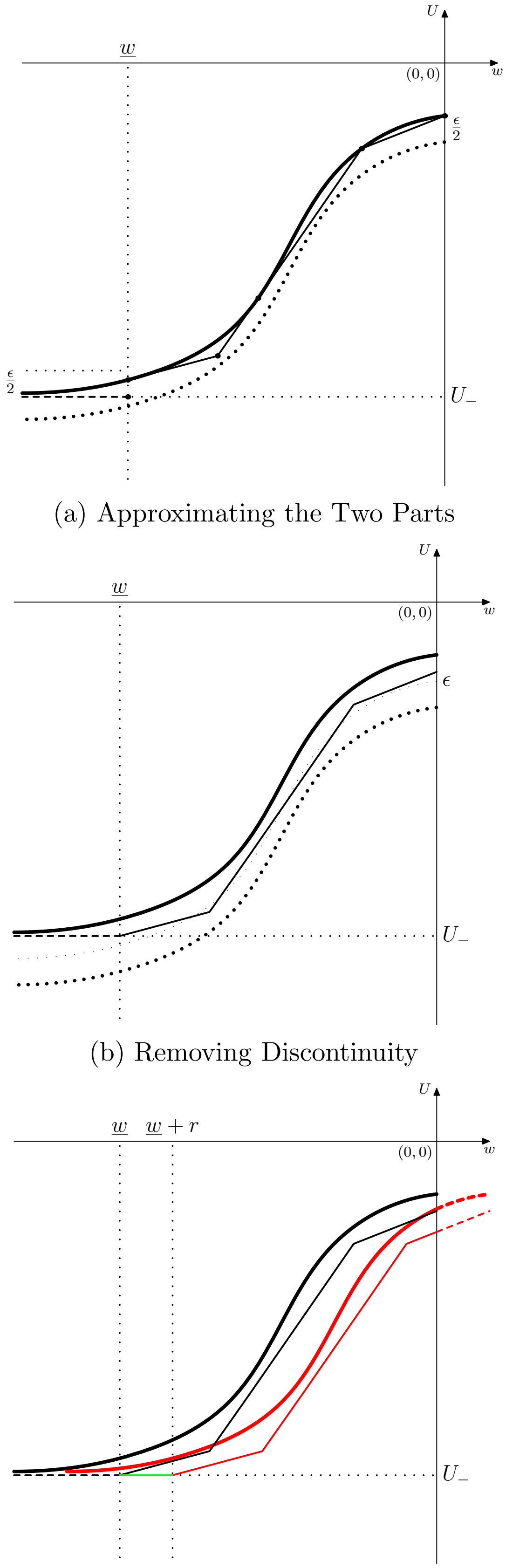
This part is illustrated in Figure 4.2(a) by solid lines.

However, there can be a discontinuity at $\underline{w}$ if we simply put these two lower approximations together. This discontinuity can be removed by shifting $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ downward by $U(\underline{w}) - U_-$. Notice that $U(\underline{w}) - \frac{\epsilon}{2} \leq U_- \leq U(\underline{w})$. Then for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \epsilon \leq U(w) + U_- - U(\underline{w}) - \frac{\epsilon}{2} \leq \underline{U}_{\frac{\epsilon}{2}}(w) + U_- - U(\underline{w}) \leq U(w) + U_- - U(\underline{w}) \leq U(w).$$

Therefore, the overall lower approximation is for all $w \leq 0$,

$$\underline{U}(w) = \begin{cases} U_-, & w \in (-\infty, \underline{w}], \\ \underline{U}_{\frac{\epsilon}{2}}(w) - U(\underline{w}) + U_-, & w \in [\underline{w}, 0], \end{cases}$$



(a) Approximating the Two Parts

(b) Removing Discontinuity

(c) Patching after Shifting

Figure 4.2: Lower PWL approximation for asymptotically constant utility functions

and the approximation error is at most ϵ. The complete lower PWL approximation is shown in Figure 4.2(b). It is monotonically nondecreasing and continuous due to our construction and properties of the methods in Section 4.2.6.

When using the value iteration procedure, we explicitly represent the functional value functions only on the finite interval $[\underline{w}, 0]$, and the functional value functions on the infinite part $(-\infty, \underline{w}]$ can be represented implicitly. Theorem 4.16 shows that the approximation error does not increase as value iteration proceeds. So we only need to consider how to perform value iteration with the initial lower bound approximation $\underline{U}(\cdot)$.

Now we consider how to perform value iteration. Suppose that the lower bound value function at step t is $V_{\underline{U}}^t(s, \cdot)$ for all states $s \in S$, and that it is represented explicitly on $[\underline{w}, 0]$. We first show that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \underline{w}]$, it holds that

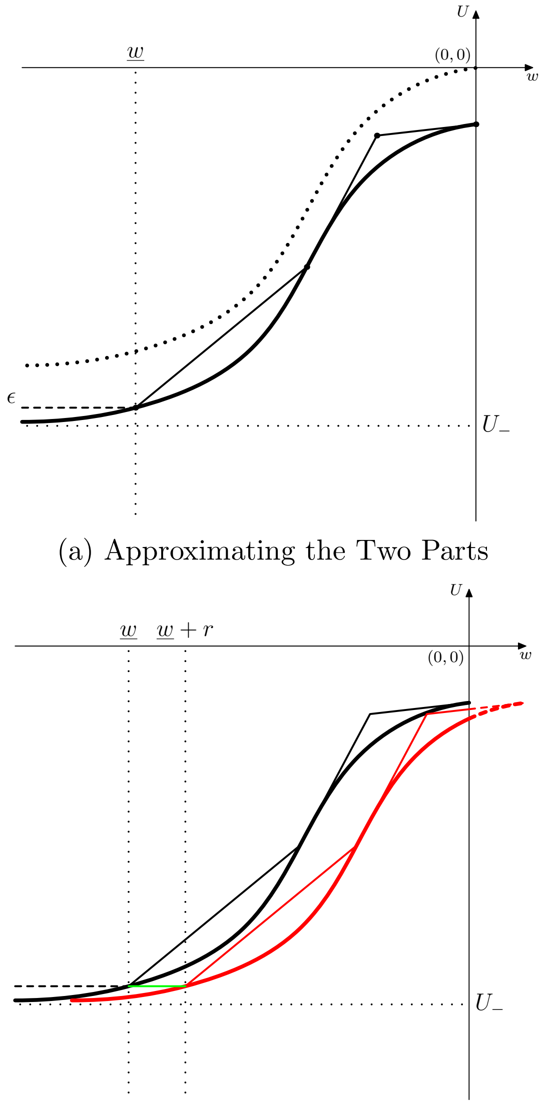
$$V_{\underline{U}}^t(s, w) = U_-.$$

This claim holds trivially for $t = 0$. Suppose it holds for t. Consider the case for $t + 1$. For all $s \in S$ and all $w \in (-\infty, \underline{w}]$, we have

$$\begin{aligned}
V_{\underline{U}}^{t+1}(s, w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) V_{\underline{U}}^t(s', w + r(s, a, s')) \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) U_- && \triangleright \quad r(s, a, s') \leq 0 \text{ and induction hypothesis} \\
&= U_-.
\end{aligned}$$

Therefore, we can always use U_- as a lower approximation of the part of the value function on $(-\infty, \underline{w}]$.

The key operation in the functional version of value iteration is the shifting operation. Since the model is negative, the $\lll$ operation shifts the functional value functions to the right, or shifts them to the left by a negative amount. For simplicity, let $V = V_{\underline{U}}^t(s, \cdot)$ be the PWL value function that is represented explicitly only on $[\underline{w}, 0]$. If we shift V to the left by an amount $r < 0$ (or equivalently, to the right by $|r|$), we obtain a PWL function $V{\lll}r$ that is explicitly represented on $[\underline{w} - r, -r]$. We can trim $V{\lll}r$ to $[\underline{w} - r, 0]$, but we also need to determine its value on the gap of the interval $[\underline{w}, \underline{w} - r]$ to make the approximation explicit. The value on this part is simply U_- because of the result we just showed. The



(a) Approximating the Two Parts

(b) Patching after Shifting

Figure 4.3: Upper PWL approximation for asymptotically constant utility functions

shifting and patching are illustrated in Figure 4.2(c), where the red lines are functions after shifting and the green segment is the patch.

Now we consider an upper approximation $\overline{U}(\cdot)$ of $U(\cdot)$, as shown in Figure 4.3. Similar to the lower approximation, for a given error $\epsilon > 0$, we can determine $\overline{w}$ such that for all $w \leq \overline{w}$, it holds that

$$U_- \leq U(w) \leq U(\overline{w}) \leq U_- + \epsilon \leq U(w) + \epsilon.$$

Then on the finite interval $[\overline{w}, 0]$, we can approximate $U(\cdot)$ from above with error ϵ, denoted as $\overline{U}_\epsilon(\cdot)$ where $\overline{U}_\epsilon(\overline{w}) = U(\overline{w})$ and $\overline{U}_\epsilon(0) = U(0)$, such that for all $w \in [\overline{w}, 0]$, it holds that

$$U(w) \leq \overline{U}_\epsilon(w) \leq U(w) + \epsilon.$$

Therefore, the overall upper approximation is for all $w \leq 0$,

$$\overline{U}(w) = \begin{cases} U(\overline{w}), & w \in (-\infty, \overline{w}], \\ \\ \overline{U}_\epsilon(w), & w \in [\overline{w}, 0], \end{cases}$$

and the approximation error is at most ϵ, as illustrated in Figure 4.3(a). The complete upper PWL approximation has to be monotonically nondecreasing and continuous, which holds due to our construction.

Similarly, we can perform value iteration with the upper PWL approximation. An argument similar to the case of the lower approximation shows that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \overline{w}]$, it holds that $V^t_{\overline{U}}(s, w) = U(\overline{w})$, and thus we can patch the gap with $U(\overline{w})$ in the same way as in the case of the lower approximation, as shown in Figure 4.3(b).

4.3.4.2 Asymptotically Linear Utility Functions

A utility function is asymptotically linear as $w \to -\infty$ if there exists $k_- > 0$ and b_- such that

$$\lim_{w \to -\infty} \big(U(w) - k_- w\big) = b_-.$$

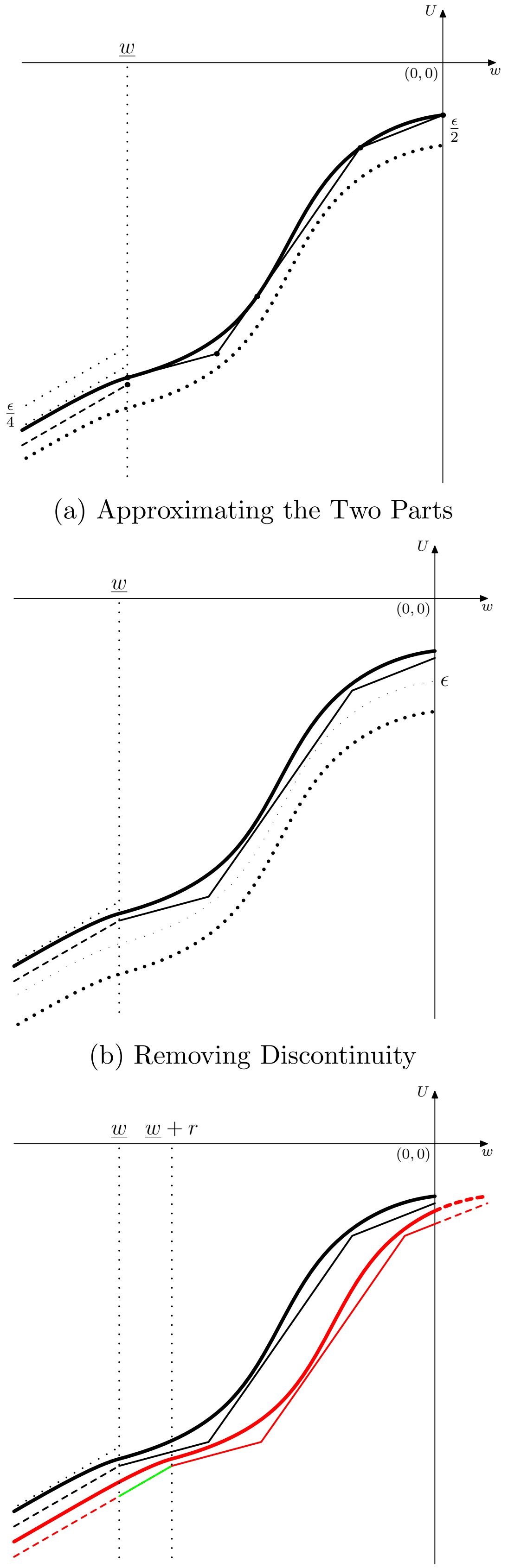
Here we assume $k_- > 0$ since otherwise the utility function is reduced to an asymptotically constant utility function.

We consider a lower approximation $\underline{U}(\cdot)$ to $U(\cdot)$, as illustrated in Figure 4.4. Similar to the case of asymptotically constant utility functions, for a given approximation error ϵ, we first determine $\underline{w} < 0$ such that for all $w \leq \underline{w}$,

$$k_- w + b_- - \frac{\epsilon}{4} \leq U(w) \leq k_- w + b_- + \frac{\epsilon}{4}.$$

Then it follows that

$$U(w) - \frac{\epsilon}{2} \leq k_- w + b_- - \frac{\epsilon}{4} \leq U(w).$$



(a) Approximating the Two Parts

(b) Removing Discontinuity

(c) Patching after Shifting

Figure 4.4: Lower PWL approximation for asymptotically linear utility functions

Therefore, we can use the linear function $k_- w + b_- - \frac{\epsilon}{4}$ to approximate $U(w)$ from below on the infinite interval $(-\infty, \underline{w}]$, and the approximation has an error of at most $\frac{\epsilon}{2}$. This part is illustrated by the dashed segment in Figure 4.4(a). We can also approximate $U(w)$ on the finite interval $[\underline{w}, 0]$ by an error of at most $\frac{\epsilon}{2}$, denoted as $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ where $\underline{U}_{\frac{\epsilon}{2}}(\underline{w}) = U(\underline{w})$ and $\underline{U}_{\frac{\epsilon}{2}}(0) = U(0)$ such that for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \frac{\epsilon}{2} \leq \underline{U}_{\frac{\epsilon}{2}}(w) \leq U(w).$$

This part is illustrated by solid lines in Figure 4.4(a). Again, if we put these two lower approximations together, there can be a discontinuity at $\underline{w}$. These problems can be easily remedied by shifting $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ downwards by $U(\underline{w}) - k_- \underline{w} - b_- + \frac{\epsilon}{4}$. Then for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \epsilon \leq \underline{U}_{\frac{\epsilon}{2}}(w) - U(\underline{w}) + k_- \underline{w} + b_- - \frac{\epsilon}{4} \leq U(w),$$

since $U(\underline{w}) - k_- \underline{w} - b_- + \frac{\epsilon}{4} \leq \frac{\epsilon}{2}$. Therefore, the overall lower approximation is for all $w \leq 0$,

$$\underline{U}(w) = \begin{cases} k_- w + b_- - \frac{\epsilon}{4}, & w \in (-\infty, \underline{w}], \\ \underline{U}_{\frac{\epsilon}{2}}(w) - U(\underline{w}) + k_- \underline{w} + b_- - \frac{\epsilon}{4}, & w \in [\underline{w}, 0], \end{cases}$$

and the approximation error is at most ϵ. The complete approximation is shown in Figure 4.4(b).

We can perform value iteration with $\underline{U}(\cdot)$ as the initial value function. First, we consider the infinite interval $(-\infty, \underline{w}]$. We show that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \underline{w}]$, it holds that

$$V^t_{\underline{U}}(s, w) = k_- w + k_- v^t(s) + b_- - \frac{\epsilon}{4},$$

where $v^0(s) = 0$ and

$$v^t(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a)[r(s, a, s') + v^{t-1}(s')].$$

The result holds trivially for $t = 0$. Suppose it holds for t. Consider the case for $t + 1$. For all $s \in S$ and all $w \in (-\infty, \underline{w}]$, we have

$$V^{t+1}_{\underline{U}}(s, w) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) V^t_{\underline{U}}(s', w + r(s, a, s'))$$

$$
\begin{aligned}
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(k_-(w + r(s,a,s')) + k_- v^t(s') + b_- - \frac{\epsilon}{4} \right) \\
&= k_- w + k_- \left(\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a)(r(s,a,s') + v^t(s')) \right) + b_- - \frac{\epsilon}{4} \\
&= k_- w + k_- v^{t+1}(s) + b_- - \frac{\epsilon}{4}.
\end{aligned}
$$

Therefore, the result holds. From Section 2.3.3, we see that $v^t(s)$ is the sequence of values under the MER objective obtained using value iteration on the original problem with 0 as the initial values.

We next consider the shifting operation. We know from the previous section that we need to patch the gap of the value function on the interval $[\underline{w}, \underline{w} - r]$ where $r < 0$. The above result shows that for step t and state s, we only need to add a segment to the PWL function representing the current value function, and this segment connects $\left(\underline{w}, k_- \underline{w} + k_- r + k_- v^t(s) + b_- - \frac{\epsilon}{4}\right)$ and $\left(\underline{w} - r, k_- \underline{w} + k_- v^t(s) + b_- - \frac{\epsilon}{4}\right)$ on the w-U plane. The shifting and patching are illustrated in Figure 4.4(c), where the red lines are functions after shifting and the green segment is the patch.

Now we consider an upper approximation $\overline{U}(\cdot)$ of $U(\cdot)$, as illustrated in Figure 4.5. For a given error $\epsilon > 0$, we can determine $\overline{w}$ such that for all $w \leq \overline{w}$, it holds that

$$k_- w + b_- - \frac{\epsilon}{2} \leq U(w) \leq k_- w + b_- + \frac{\epsilon}{2}.$$

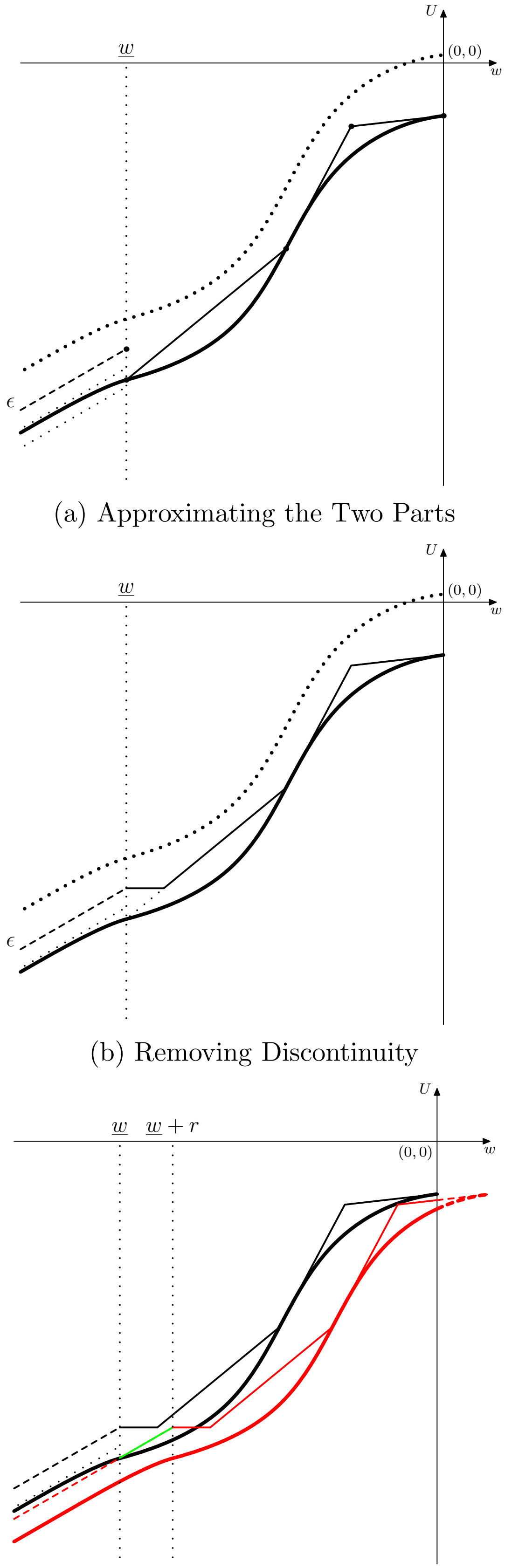
Then it follows that

$$U(w) \leq k_- w + b_- + \frac{\epsilon}{2} \leq U(w) + \epsilon.$$

Therefore, we can use the linear function $k_- w + b_- + \frac{\epsilon}{2}$ to approximate $U(w)$ for above on the infinite interval $(-\infty, \overline{w}]$, and the approximation has an error of at most ϵ. We can also approximate $U(w)$ on the finite interval $[\overline{w}, 0]$ by an error of at most ϵ , denoted as $\overline{U}_\epsilon(\cdot)$ where $\overline{U}_\epsilon(\overline{w}) = U(\overline{w})$ and $\overline{U}_\epsilon(0) = U(0)$ such that for all $w \in [\overline{w}, 0]$, it holds that

$$U(w) \leq \overline{U}_\epsilon(w) \leq U(w) + \epsilon.$$

If we put these two upper approximations together, there can be a discontinuity at $\overline{w}$ and the result can also be nonmonotonic, as shown in Figure 4.5(a). These problems



(a) Approximating the Two Parts

(b) Removing Discontinuity

(c) Patching after Shifting

Figure 4.5: Upper PWL approximation for asymptotically linear utility functions

can be solved by adjusting the upper approximation on the finite interval $[\overline{w}, 0]$ to be $\max\left(k_-\overline{w} + b_- + \frac{\epsilon}{2}, \overline{U}_\epsilon(\cdot)\right)$. The resulting PWL function still has an error of at most ϵ, since $U(\cdot)$ is nondecreasing and $k_-\overline{w} + b_- + \frac{\epsilon}{2} \leq U(\overline{w}) + \epsilon$. Therefore, the overall upper approximation is for all $w \leq 0$,

$$\overline{U}(w) = \begin{cases} k_-w + b_- + \frac{\epsilon}{2}, & w \in (-\infty, \overline{w}], \\ \max\left(k_-\overline{w} + b_- + \frac{\epsilon}{2}, \overline{U}_\epsilon(w)\right), & w \in [\overline{w}, 0], \end{cases}$$

and the approximation error is at most ϵ. The complete approximation is illustated in Figure 4.5(b).

Value iteration can be performed with the upper approximation $\overline{U}(\cdot)$. An argument similar to the case of the lower approximation shows that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \overline{w}]$, it holds that $V^t_{\overline{U}}(s, w) = k_-w + k_-v^t(s) + b_- + \frac{\epsilon}{2}$, and we can thus patch the gap with $U(\overline{w})$ in the same way as in the case of the lower approximation, as illustrated in Figure 4.5(c).

4.3.4.3 Asymptotically Exponential Utility Functions

A utility function is asymptotically exponential as $w \to -\infty$ if there exists $0 < \gamma < 1$, $k_- > 0$ and b_- such that

$$\lim_{w \to \infty} \left(U(w) + k_-\gamma^w\right) = b_-.$$

Here we assume that $k_- > 0$ and $0 < \gamma < 1$ since otherwise it is reduced to the case of asymptotically constant utility functions.

The initial lower and upper approximations for asymptotically exponential utility functions are almost exactly the same as those for asymptotically linear utility functions. Therefore, we do not include separate illustrations.

We consider a lower approximation $\underline{U}(\cdot)$ to $U(\cdot)$. Similar to the above, for a given approximation error ϵ, we first determine $\underline{w} < 0$ such that for all $w \leq \underline{w}$,

$$-k_-\gamma^w + b_- - \frac{\epsilon}{4} \leq U(w) \leq -k_-\gamma^w + b_- + \frac{\epsilon}{4}.$$

Then it follows that

$$U(w) - \frac{\epsilon}{2} \leq -k_-\gamma^w + b_- - \frac{\epsilon}{4} \leq U(w).$$

Therefore, we can use the exponential function $-k_-\gamma^w + b_- - \frac{\epsilon}{4}$ to approximate $U(w)$ from below on the infinite interval $(-\infty, \underline{w}]$, and the approximation has an error of at most $\frac{\epsilon}{2}$. We can also approximate $U(w)$ on the finite interval $[\underline{w}, 0]$ by an error of at most $\frac{\epsilon}{2}$, denoted as $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ where $\underline{U}_{\frac{\epsilon}{2}}(\underline{w}) = U(\underline{w})$ and $\underline{U}_{\frac{\epsilon}{2}}(0) = U(0)$ such that for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \frac{\epsilon}{2} \leq \underline{U}_{\frac{\epsilon}{2}}(w) \leq U(w).$$

Again, if we put these two lower approximations together, there can be a discontinuity at $\underline{w}$ and the result can also be nonmonotonic. These problems can be easily remedied by shifting $\underline{U}_{\frac{\epsilon}{2}}(\cdot)$ downwards by $U(\underline{w}) + k_-\gamma^{\underline{w}} - b_- + \frac{\epsilon}{4}$. Then for all $w \in [\underline{w}, 0]$, it holds that

$$U(w) - \epsilon \leq \underline{U}_{\frac{\epsilon}{2}}(w) - U(\underline{w}) - k_-\gamma^{\underline{w}} + b_- - \frac{\epsilon}{4} \leq U(w),$$

since $U(\underline{w}) + k_-\gamma^{\underline{w}} - b_- + \frac{\epsilon}{4} \leq \frac{\epsilon}{2}$. Therefore, the overall lower approximation is for all $w \leq 0$,

$$\underline{U}(w) = \begin{cases} -k_-\gamma^w + b_- - \frac{\epsilon}{4}, & w \in (-\infty, \underline{w}], \\ \underline{U}_{\frac{\epsilon}{2}}(w) - U(\underline{w}) - k_-\gamma^{\underline{w}} + b_- - \frac{\epsilon}{4}, & w \in [\underline{w}, 0], \end{cases}$$

and the approximation error is at most ϵ.

We can perform value iteration with $\underline{U}(\cdot)$ as the initial value function. We first show that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \underline{w}]$, it holds that

$$V^t_{\underline{U}}(s, w) = k_- v^t_{\exp}(s)\gamma^w + b_- - \frac{\epsilon}{4},$$

where $v^0_{\exp}(s) = \iota = -1$ and

$$v^t_{\exp}(s) = \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} v^{t-1}_{\exp}(s').$$

It holds trivially for $t = 0$. Suppose it holds for t. Consider the case for $t+1$. For all $s \in S$ and all $w \in (-\infty, \underline{w}]$, we have

$$\begin{aligned} V^{t+1}_{\underline{U}}(s, w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V^t_{\underline{U}}(s', w + r(s,a,s')) \\ &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(k_- v^t_{\exp}(s')\gamma^{w+r(s,a,s')} + b_- - \frac{\epsilon}{4}\right) \end{aligned}$$

$$= k_- \left(\max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \gamma^{r(s,a,s')} v_{\exp}^t(s') \right) \gamma^w + b_- - \frac{\epsilon}{4}$$

$$= k_- v_{\exp}^{t+1}(s) \gamma^w + b_- - \frac{\epsilon}{4}.$$

Therefore, the result holds. From Section 3.2, the values $v_{\exp}^t(s)$ are the sequence of values under the $\mathsf{MEU}_{\exp}$ objective obtained using value iteration on the original problem with $U_{\exp}(0) = -1$ as the initial values.

We next consider the shifting operation. We know from the previous sections that we need to patch the gap of the value function on the interval $[\underline{w}, \underline{w} - r]$ where $r < 0$. The above result shows that for step t and state s, we only need to add a segment to the lower approximation representing the current value function, and the segment is an exponential function passing points $\left(\underline{w}, k_- v_{\exp}^t(s) \gamma^{\underline{w}+r} + b_- - \frac{\epsilon}{4}\right)$ and $\left(\underline{w} - r, k_- v_{\exp}^t(s) \gamma^{\underline{w}} + b_- - \frac{\epsilon}{4}\right)$ on the w-U plane.

There remains a problem for the value iteration procedure: the value function for the newly patched interval $[\underline{w}, \underline{w} - r]$ is an exponential rather than a linear function. Thus after the patching, the explicit part of the functional value function is no longer a PWL function.

There are two solutions to this problem. The first one is to approximate this exponential segment on this interval using the sandwich method. Since this approximation introduces additional error, we need to budget our error allocation differently for the initial lower approximation to the original utility function. More specifically, we need to budget less error for the infinite part of the approximation to leave room for the additional errors from the patching. Since we consider negative models, the functional value function can only be shifted to the right. Therefore, the newly introduced error will not affect the error on the interval $[\underline{w} - r, 0]$.

The second solution is to work directly with the mixed exponential and linear segments of the approximation. The benefit of this solution is that no additional error will be introduced. Because of the weighted summation operation of the value update rule, we need to deal with piecewise functions, each of which is the summation of an exponential and a linear function. Now we briefly discuss this approach. There are a lot of similarities to the case of PWL functions. A minor difference from the case of PWL functions is that we represent the piecewise one-switch functions using three parameters, corresponding to the coefficients of the linear, exponential, and constant terms, respectively. Suppose that the three parameters are C, D, E. They represent a function $Cw + D\gamma^w + E$, referred to as one-switch functions, since they are related to planning with one-switch utility functions (see Section 4.4). Similar to the PWL case, we only need to consider scaling, shifting, addition, and maximization over a fixed interval. With the triple parameter representation, scaling, shifting, and addition can be done in the same way as PWL functions. The maximization operation is slightly different. Similar to the PWL case, the maximization operation for two piecewise one-switch functions can also be reduced to the maximization operation for two one-switch functions on a finite interval. There can be (at most) two intersections for two one-switch functions on a finite interval. The maximization operation then need to take this possibility into account. An illustration of this approach is shown in Section 4.3.5.2.

An upper approximation can be obtained in a similar way to that for asymptotically linear utility functions. For a given error $\epsilon > 0$, we can determine $\overline{w}$ such that for all $w \leq \overline{w}$, it holds that

$$-k_-\gamma^w + b_- - \frac{\epsilon}{2} \leq U(w) \leq -k_-\gamma^w + b_- + \frac{\epsilon}{2}.$$

Then it follows that

$$U(w) \leq -k_-\gamma^w + b_- + \frac{\epsilon}{2} \leq U(w) + \epsilon.$$

Therefore, we can use the exponential function $-k_-\gamma^w + b_- + \dfrac{\epsilon}{2}$ to approximate $U(w)$ from above on the infinite interval $(-\infty, \overline{w}]$, and the approximation has an error of at most ϵ. We can also approximate $U(w)$ on the finite interval $[\overline{w}, 0]$ with an error of at most ϵ, denoted as $\overline{U}_\epsilon(\cdot)$ where $\overline{U}_\epsilon(\overline{w}) = U(\overline{w})$ and $\overline{U}_\epsilon(0) = U(0)$ such that for all $w \in [\overline{w}, 0]$, it holds that

$$U(w) \leq \overline{U}_\epsilon(w) \leq U(w) + \epsilon.$$

If we put these two upper approximations together, there can be a discontinuity at $\overline{w}$ and the result can also be nonmonotonic. These problems can be solved by adjusting the upper approximation on the finite interval $[\overline{w}, 0]$ to be $\max\left(-k_-\gamma^{\overline{w}} + b_- + \dfrac{\epsilon}{2}, \overline{U}_\epsilon(\cdot)\right)$. The resulting PWL function still has an error of at most ϵ, since $U(\cdot)$ is nondecreasing and

$-k_{-}\gamma^{\overline{w}} + b_{-} + \dfrac{\epsilon}{2} \leq U(\overline{w}) + \epsilon$. Therefore, the overall upper approximation is for all $w \leq 0$,

$$\overline{U}(w) = \begin{cases} -k_{-}\gamma^{w} + b_{-} + \dfrac{\epsilon}{2}, & w \in (-\infty, \overline{w}], \\ \max\left(-k_{-}\gamma^{\overline{w}} + b_{-} + \dfrac{\epsilon}{2}, \overline{U}_{\epsilon}(w)\right), & w \in [\overline{w}, 0], \end{cases}$$

and the approximation error is at most ϵ.

Value iteration can be performed with the upper approximation $\overline{U}(\cdot)$. An argument similar to the case of the lower approximation shows that for all $t \in \mathbb{N}$, all $s \in S$, and all $w \in (-\infty, \overline{w}]$, it holds that $V^{t}_{\overline{U}}(s, w) = -k_{-}v^{t}_{\exp}\gamma^{w} + b_{-} + \dfrac{\epsilon}{2}$, and thus we can patch the gap with $U(\overline{w})$ in the same way as in the case of the lower approximation.

4.3.5 Example: Deadline Utility Functions

We now illustrate value iteration with functional value functions using deadline utility functions and the painted-blocks problem. Deadline utility functions express the preference that completing the task by the given deadline is preferred over not completing the task by the deadline, and completing the task by the deadline with a higher probability is preferred over completing the task by the deadline with a lower probability (Haddawy and Hanks, 1992).

Deadlines can be hard or soft. With a hard deadline, it is useless to complete the task after the deadline. With a soft deadline, the utility of completing the task after the deadline decreases gradually. We demonstrate how the functional version of value iteration can be used to solve planning problems with both hard and soft deadlines, using a painted-blocks problem with a different initial state, while the rest of the problem is unchanged from Chapter 1. With the new initial state, we are able to show that there are different optimal policies with different utility functions. The new initial state is shown in Figure 4.6, which also shows the goal configuration for convenience. This example is also used later with one-switch utility functions (see Section 4.4.3). Recall that each action takes some time to perform, and thus the planning problem is modeled as a negative model.

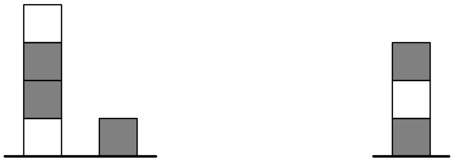


Figure 4.6: A painted-blocks problem for general risk-sensitive utility functions

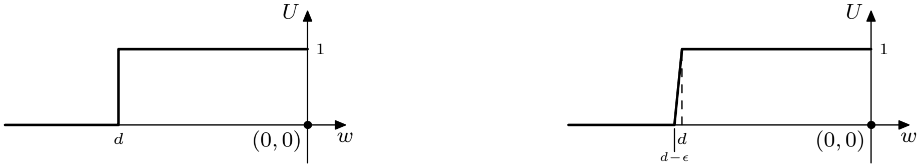


Figure 4.7: A hard deadline utility function and its modification for value iteration

4.3.5.1 Hard Deadlines

A hard deadline can be modeled as a step function. If the deadline is $d < 0$, the hard-deadline utility function is

$$U_{\text{hdl}}(w) = \begin{cases} 1, & w \geq d, \\ 0, & w < d. \end{cases}$$

This is a PWL function except that there is a discontinuity. In our earlier discussion, we assumed that the PWL functions be continuous. We can remove the discontinuity by modifying the utility function slightly. Let $\epsilon > 0$ be a small number such that $(d-\epsilon, d) \cap W = \varnothing$. Then we can use the modified hard-deadline utility function

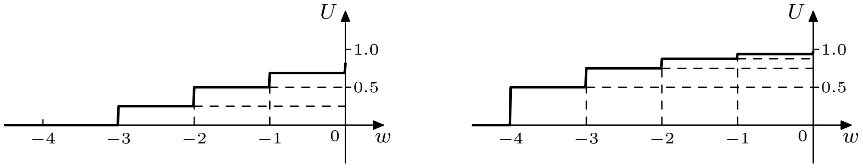
$$U_{\text{hdl}'}(w) = \begin{cases} 1, & w \geq d, \\ \frac{1}{\epsilon}(x-d)+1, & d-\epsilon \geq w < d, \\ 0, & w < d-\epsilon. \end{cases}$$

and for all $w \in W$, $U_{\text{hdl}}(w) = U_{\text{hdl}'}(w)$.

With a hard deadline function, the value of a state for a policy is also the probability of reaching the goal state before the deadline starting from this state under the control of this policy. Obviously, there exists a policy that can complete the task with a total cost of 7 (a total reward of -7): remove the top block of the tower of four blocks and paint the two bottom blocks of the tower. Table 4.1 lists the different optimal values for different values of d obtained by using value iteration with functional value functions.

Table 4.1: Optimal values (probabilities of success) with hard deadlines

d	$[0,-2)$	$[-2,-3)$	$[-3,-4)$	$[-4,-5)$	$[-5,-6)$	$[-6,-7)$	$[-7,-\infty)$
v^*_{hdl}	0	0.25	0.5	0.6875	0.8125	0.890625	1.0



(a) {`WBBW,B`} and {`WBB,B,W`} (b) {`WBB,BW`} and {`BW,WB,B`}

Figure 4.8: Optimal functional value functions of relevant states for the hard deadline utility function

Now we look at the results for $d = -5$ in more detail. We first consider the functional value functions, then retrieve an SD-optimal policy for the augmented model, and lastly obtain an aSD-optimal policy for the original model. We consider only relevant states starting from the initial state, since not all states are involved in the optimal policy for the original model.[4] For goal states, the functional value function is always $U_{\text{hdl}}(\cdot)$. The functional value functions for states relevant to the initial state {`WBBW,B`} are shown in Figure 4.8. Based on the value functions, we can retrieve an optimal policy for the augmented model, as shown in Figure 4.9. The actions are optimal under particular conditions, which are also shown in the figure. Notice that these conditions do not cover all possible $w \in W$. If the agent is in a state and its accumulated wealth is not covered by the condition, any action is optimal since no goal state can be reached within the deadline. Thus such conditions and actions are omitted.

To obtain an optimal policy for the original model, we only need to restrict the optimal policy for the augmented model to those wealth levels that can be achieved starting from the initial state and an initial wealth of zero. This can be done using a forward search. In this way, a corresponding optimal policy for the original model is shown in Figure 4.10, where the difference is that the action for state {`WBBW,B`} does not require a condition, since

[4] Strictly speaking, not all states are involved in the "interesting" part of the optimal policy. If the total reward is less than d, any action is optimal, and thus all states are involved. But states and actions involved this way are the "boring" part of the policy.

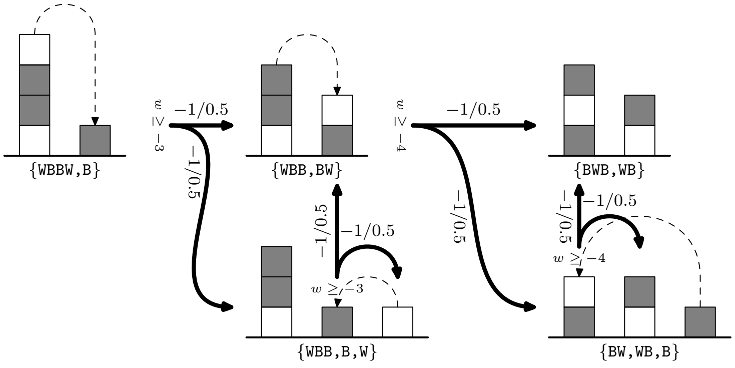


Figure 4.9: An SD-optimal policy for the augmented model with the hard deadline utility function

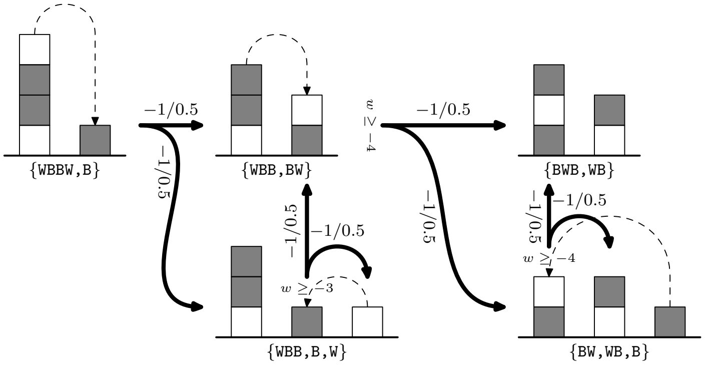


Figure 4.10: The aSD-optimal policy for the original model with the hard deadline utility function

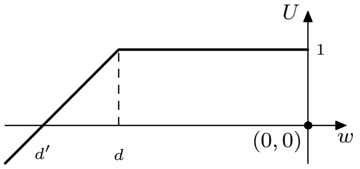


Figure 4.11: Linearly soft deadline utility function

the initial wealth level is zero. The optimal policy indicates that the agent keeps trying moving actions until it is impossible to reach a goal state within the deadline.

4.3.5.2 Soft Deadlines

For the hard deadline utility function, any action is equally good (or bad) if no goal state can be reached within the deadline. It is often desirable for the agent to complete the task even if the deadline is missed. Soft deadline utility functions can be used to achieve this effect. For soft deadline utility functions, the utility decreases gradually if the deadline is missed. Common types of soft deadline utility functions include linearly and exponentially soft deadline utility functions.

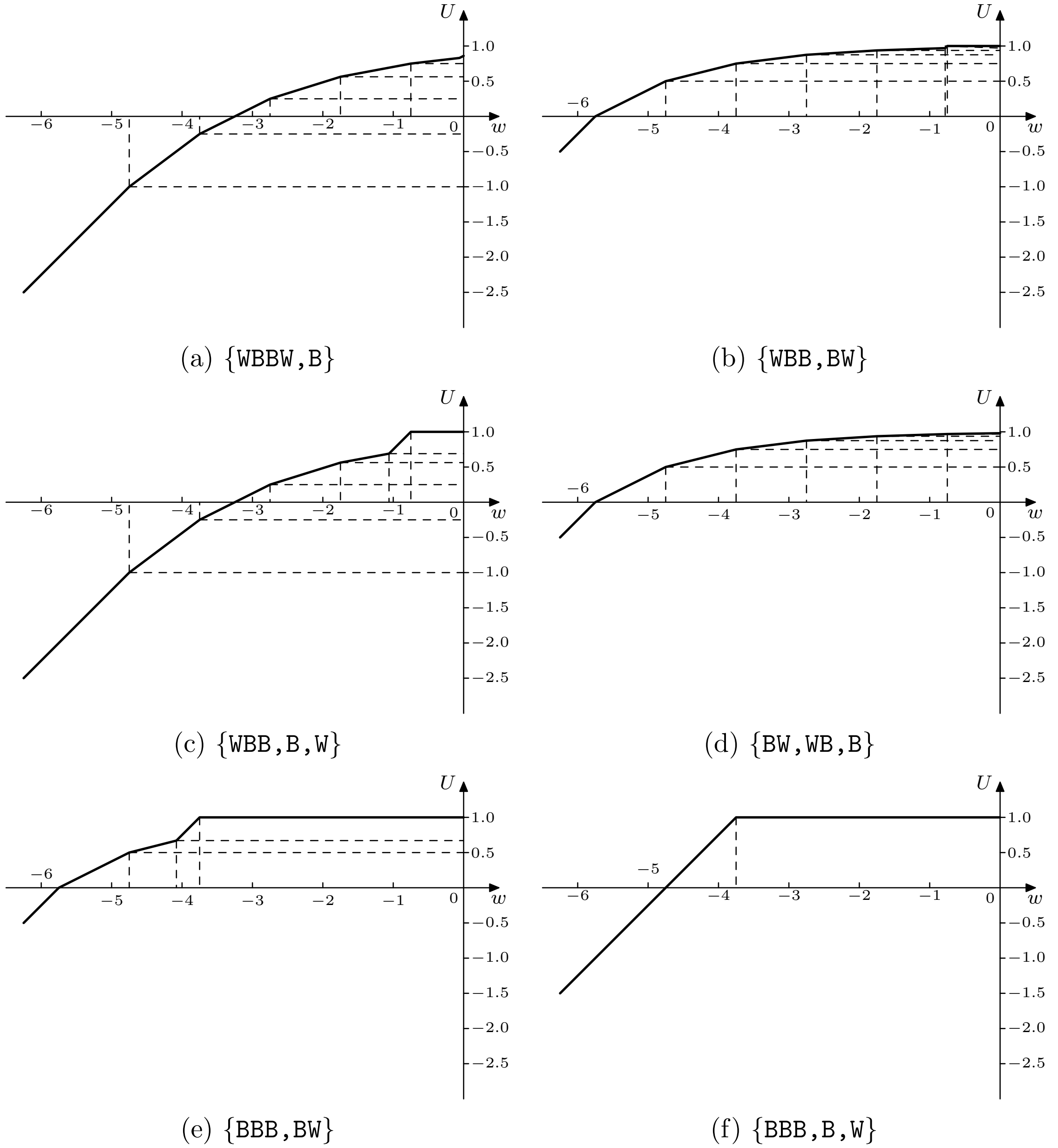
4.3.5.2.1 Linearly Soft Deadlines

A linearly soft deadline utility function is defined as

$$U_{\text{sdl-l}}(w) = \begin{cases} 1, & w \geq d, \\ \dfrac{w - d'}{d - d'}, & w < d, \end{cases}$$

so that $U_{\text{sdl-l}}(d') = 0$, as shown in Figure 4.11. This function is a PWL function and value iteration can be directly applied.

For illustration purposes, we are interested in a linearly soft deadline utility function with which the optimal policy is qualitatively different from the case of a hard deadline utility function, and also different from the case of a linear utility function. This is the case if we let $d = -6.75$ and $d' = -7.75$. The functional value functions are also PWL functions, as shown in Figure 4.12. The breakpoints of the value functions are listed in Table 4.2. The leftmost part of the value functions is always a linear function with a slope of 1.0.



(a) {WBBW,B}

(b) {WBB,BW}

(c) {WBB,B,W}

(d) {BW,WB,B}

(e) {BBB,BW}

(f) {BBB,B,W}

Figure 4.12: Optimal functional value functions of relevant states for the linearly soft deadline utility function

Table 4.2: Breakpoints of the value functions from Figure 4.12

{WBBW,B}	{WBB,BW}	{WBB,B,W}	{BW,WB,B}	{BBB,BW}	{BBB,B,W}
(−4.75, −1.00)	(−5.75, 0.00)	(−4.75, −1.00)	(−5.75, 0.00)	(−5.75, 0.00)	(−3.75, 1.00)
(−3.75, −0.25)	(−4.75, 0.50)	(−3.75, −0.25)	(−4.75, 0.50)	(−4.75, 0.50)	(0.00, 1.00)
(−2.75, 0.25)	(−3.75, 0.75)	(−2.75, 0.25)	(−3.75, 0.75)	(−4.08, 0.67)	
(−1.75, 0.56)	(−2.75, 0.86)	(−1.75, 0.56)	(−2.75, 0.86)	(−3.75, 1.00)	
(−0.75, 0.75)	(−1.75, 0.94)	(−1.06, 0.69)	(−1.75, 0.94)	(0.00, 1.00)	
(−0.06, 0.83)	(−0.78, 0.97)	(−0.75, 1.00)	(−0.75, 0.97)		
(0.00, 0.86)	(−0.75, 1.00)	(0.00, 1.00)	(0.00, 0.98)		
	(0.00, 1.00)				

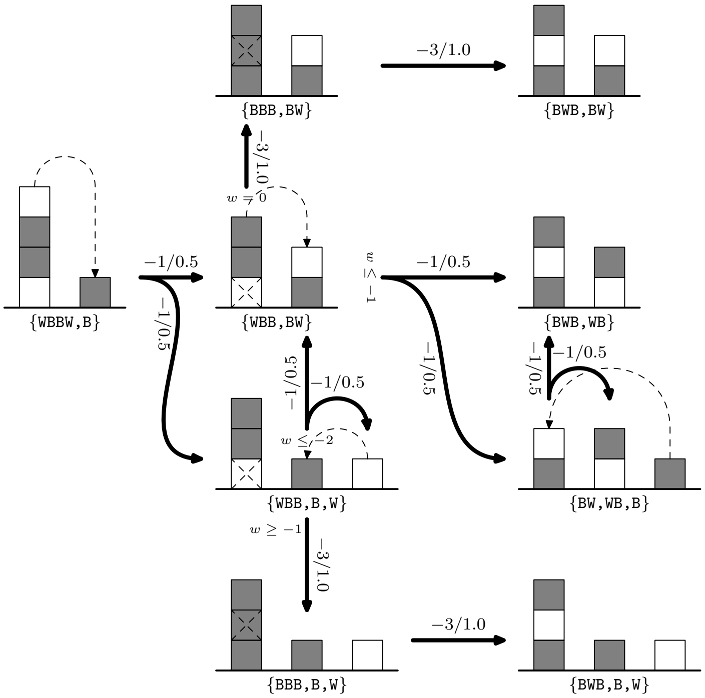


Figure 4.13: An SD-optimal policy for the augmented model with the linearly soft deadline utility function

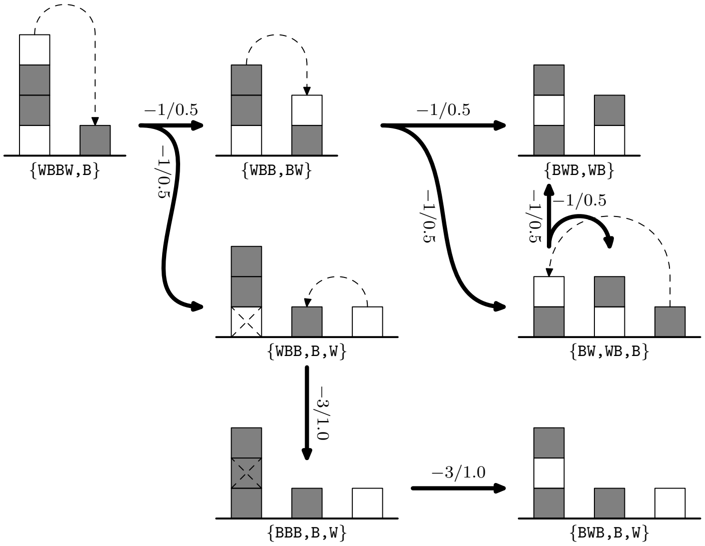


Figure 4.14: The aSD-optimal policy for the original model with the linearly soft deadline utility function

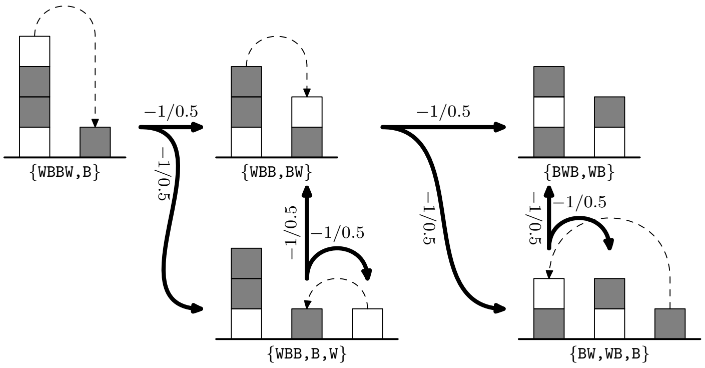


Figure 4.15: The SD-optimal policy for the original model with a linear utility function

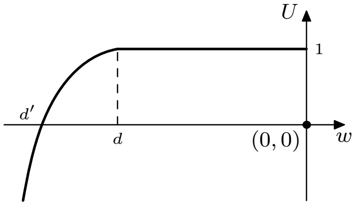
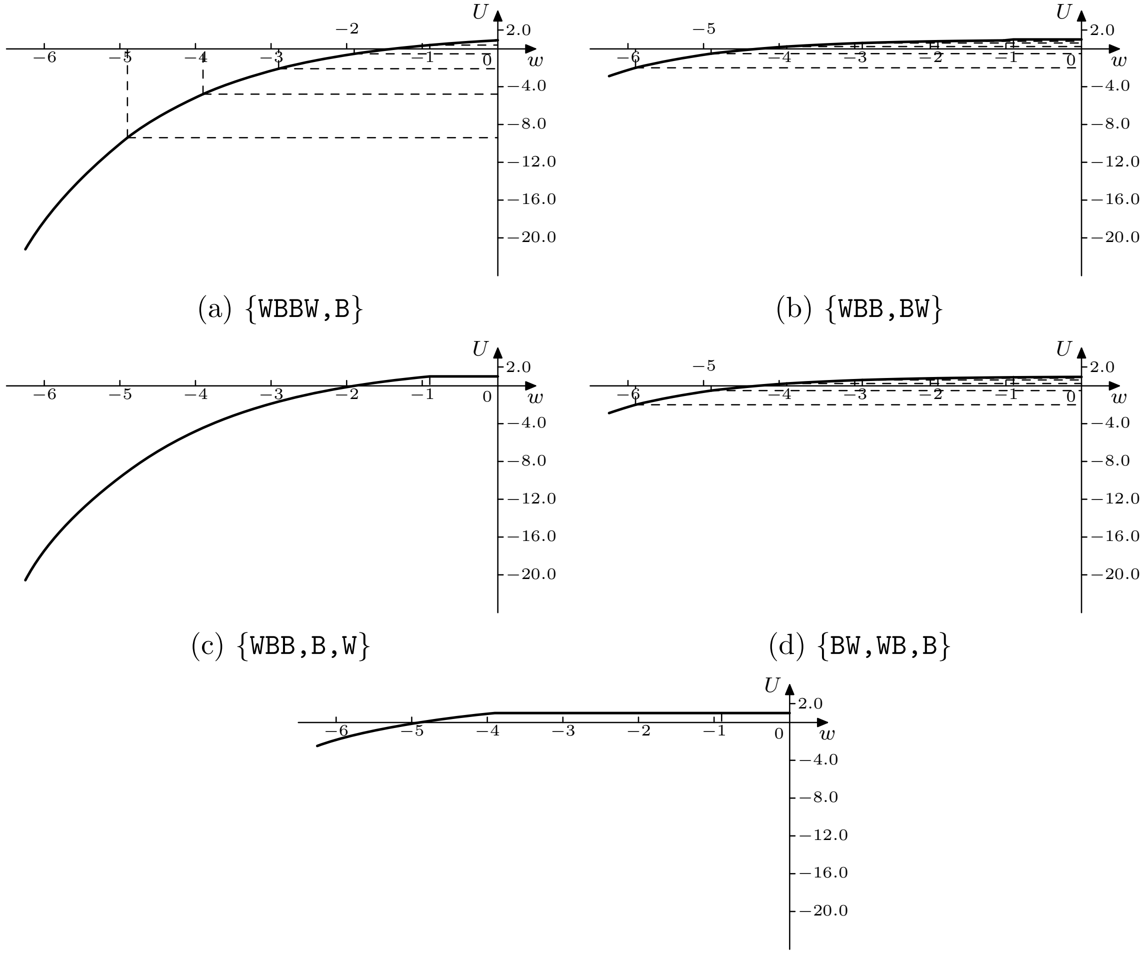


Figure 4.16: Exponentially soft deadline utility function

Based on the value functions, we can retrieve an optimal policy for the augmented model, as shown in Figure 4.13. The corresponding optimal policy for the original model, as shown in Figure 4.14, is simpler, since unnecessary branches, for example, $w = 0$ in state {WBB,BW} and $w \leq -2$ in state {WBB,B,W}, can be removed. Following the optimal policy, the agent tries moving actions until a goal state is reached, but if the first moving action ({WBBW,B} → {WBB,BW}) fails, the agent switches to the painting strategy. The resulting policy also differs from the optimal policy with a linear utility function, which uses only moving actions, as shown in Figure 4.15.

Since the linearly soft deadline utility function has a tail that is a linear utility function, the optimal policies with these two utility functions are related. This relationship is best explained by comparing the SD-optimal policy for the augmented model under the MEU model and the SD-optimal policy under the MER model. When w is large enough in magnitude, the SD-optimal policy for the augmented model is reduced to the SD-optimal policy under the MER model. All the differences between these two policies are due to the nonlinear part of the utility function, which takes effect when w is small in magnitude.



(a) {WBBW,B}

(b) {WBB,BW}

(c) {WBB,B,W}

(d) {BW,WB,B}

(e) {BBB,BW} and {BBB,B,W}

Figure 4.17: Optimal functional value functions of relevant states for the exponentially soft deadline utility function

Table 4.3: Breakpoints of the value functions from Figure 4.17

				{BBB,BW}
{WBBW,B}	{WBB,BW}	{WBB,B,W}	{BW,WB,B}	{BBB,B,W}
−0.9734	−0.2210	−0.9472	−0.2210	−0.2046
(−4.90, −9.40)	(−5.90, −2.00)	(−0.90, 1.00)	(−5.90, −2.00)	(−3.90, 1.00)
(−3.90, −4.79)	(−4.90, −0.50)	(0.00, 1.00)	(−4.90, −0.50)	(0.00, 1.00)
(−2.90, −2.10)	(−3.90, 0.25)		(−3.90, 0.25)	
(−1.90, −0.52)	(−2.90, 0.63)		(−2.90, 0.63)	
(−0.90, 0.41)	(−1.90, 0.81)		(−1.90, 0.81)	
(−0.03, 0.90)	(−1.03, 0.90)		(−0.90, 0.91)	
(0.00, 0.92)	(−0.90, 1.00)		(0.00, 0.95)	
	(0.00, 1.00)			

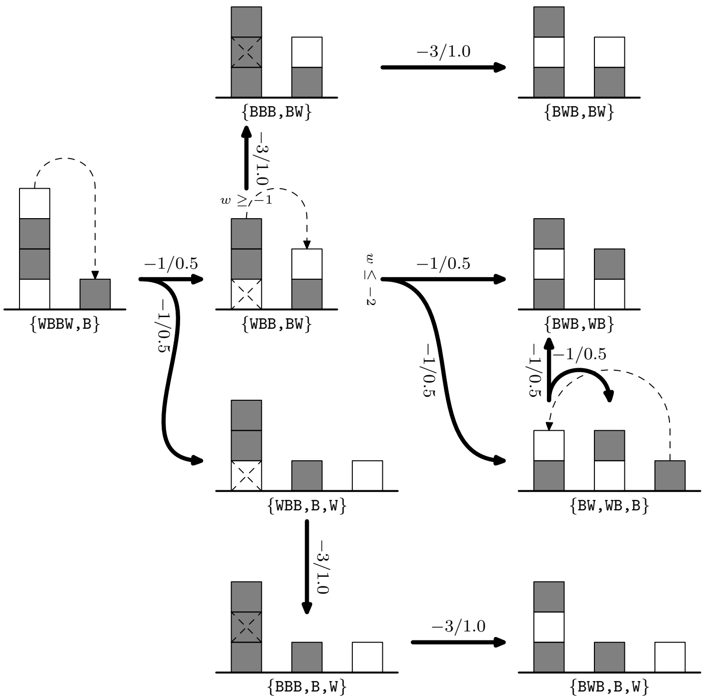


Figure 4.18: An SD-optimal policy for the augmented model with the exponentially soft deadline utility function

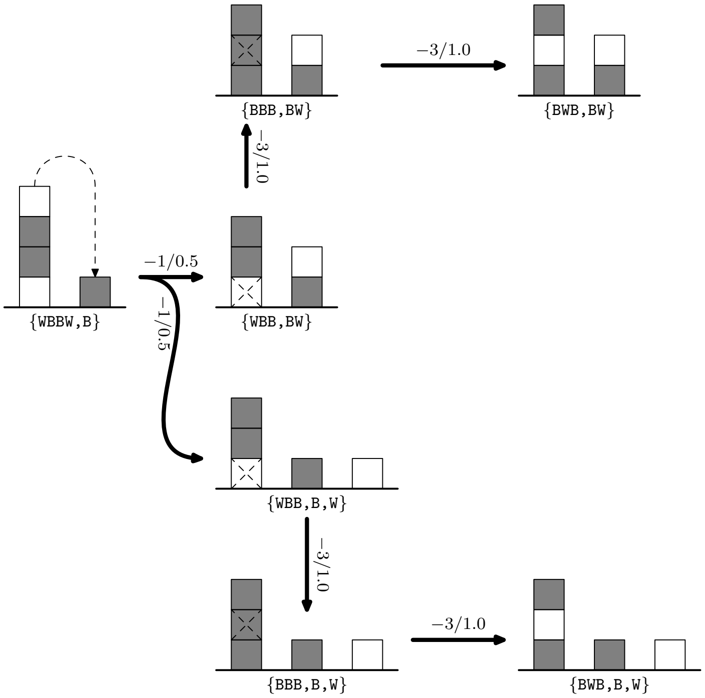


Figure 4.19: The aSD-optimal policy for the original model with the exponentially soft deadline utility function

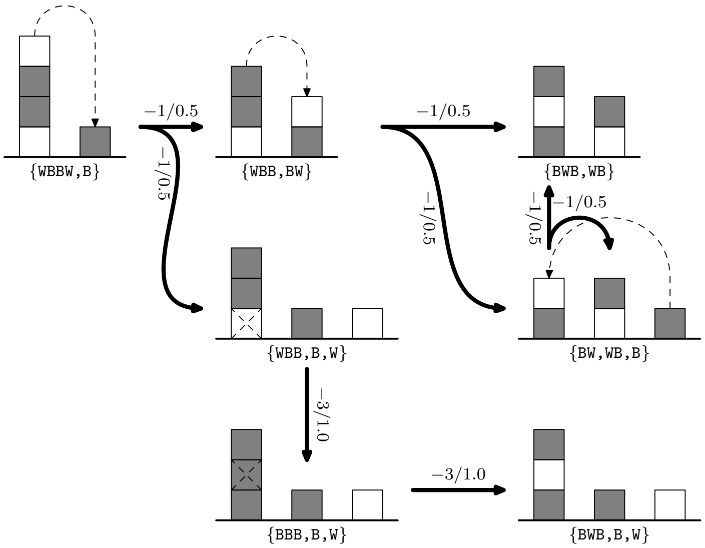


Figure 4.20: The SD-optimal policy for the original model with an exponential utility function

4.3.5.2.2 Exponentially Soft Deadlines

An exponentially soft deadline utility function is defined as

$$U_{\text{sdl-e}}(w) = \begin{cases} 1, & w \geq d, \\ \dfrac{\gamma^w - \gamma^{d'}}{\gamma^d - \gamma^{d'}}, & w < d, \end{cases}$$

so that $U_{\text{sdl-e}}(d') = 0$, as shown in Figure 4.16. The utility function is not a PWL function, but is asymptotically exponential. We therefore use the method outlined in Section 4.3.4.3, using piecewise mixed exponential and linear functions (in fact, the linear term always has a zero coefficient).

Again, we are interested in an exponentially soft deadline utility function with which the optimal policy is qualitatively different from the case of a hard deadline utility function, and also different from the case of an exponential utility function with the same γ. This is the case if we let $\gamma = 0.60$, $d = -6.90$, and $d' = -7.90$. The functional value functions are piecewise exponential, as shown in Figure 4.17. The breakpoints of the value functions are listed in Table 4.3. The numbers listed on the top are the coefficient of the exponential terms for the leftmost segment.

Based on the value functions, we can retrieve an optimal policy for the augmented model, as shown in Figure 4.18. The corresponding optimal policy for the original model is shown

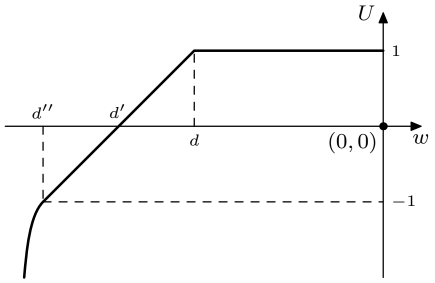


Figure 4.21: Mixed soft deadline utility function

in Figure 4.19, where unnecessary branches, for example, $w \leq -2$ in state {`WBB,BW`}, are removed. Following the optimal policy, the agent uses only painting actions. The resulting policy also differs from the optimal policy with an exponential utility function with the same γ, which uses both painting and moving actions, as shown in Figure 4.20.

Similar to linearly soft deadline utility functions, the optimal policy with the exponentially soft deadline utility function is related to that with an exponential utility function. When w is large enough in magnitude, the SD-optimal policy for the augmented model is reduced to the SD-optimal policy under the $\mathsf{MEU}_{\text{exp}}$ model. All the differences between these two policies are due to the non-exponential part of the utility function, which takes effect when w is small in magnitude.

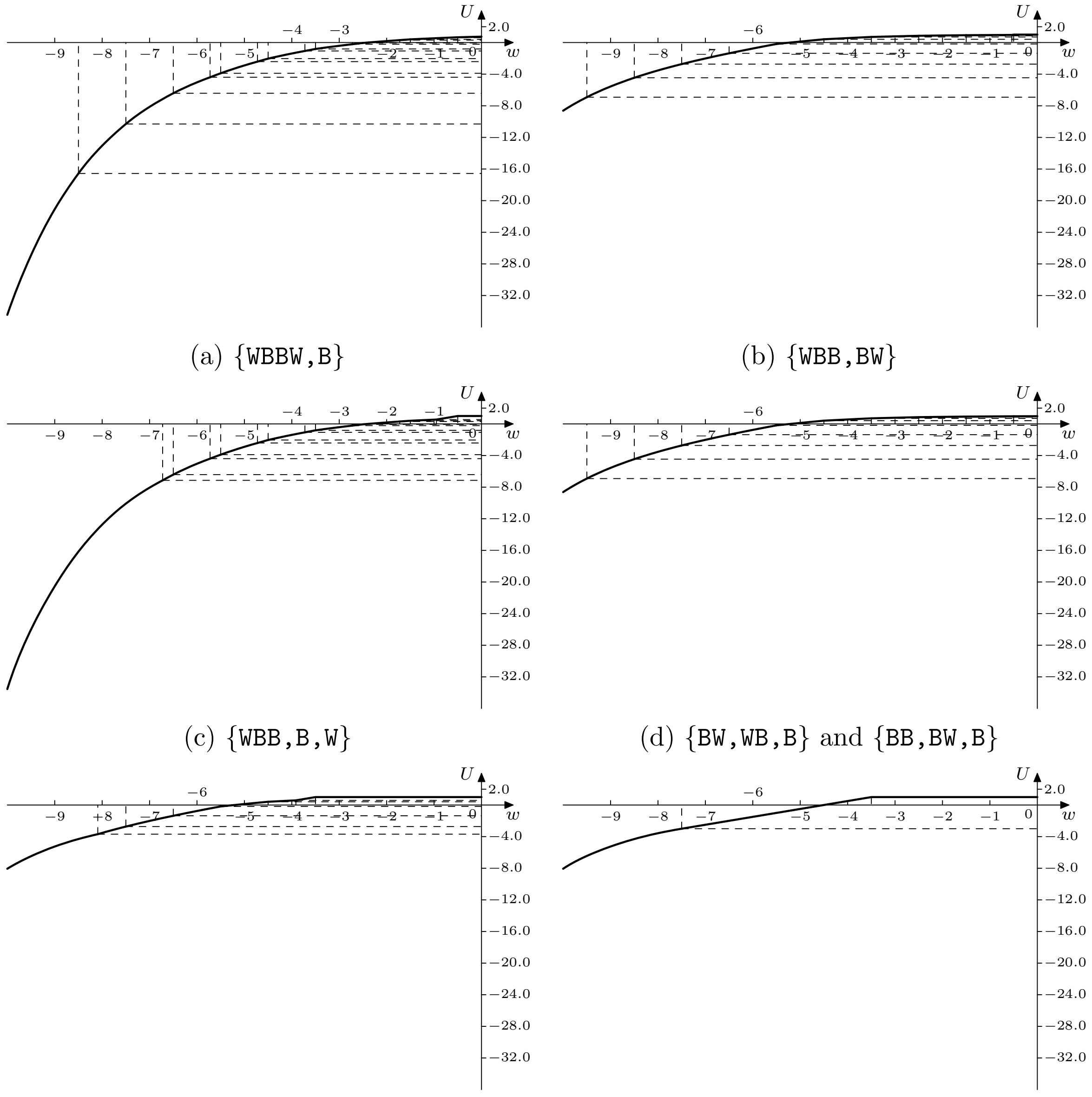
4.3.5.2.3 Mixed Soft Deadlines

We consider a mixed soft deadline utility function, defined as

$$U_{\text{sdl-e-l}}(w) = \begin{cases} 1, & w \geq d, \\ \dfrac{w - d'}{d - d'}, & d' \leq w < d, \\ \dfrac{\gamma^{w-d''} + (d'' - d')\ln\gamma - 1}{(d - d')\ln\gamma}, & w < d'', \end{cases}$$

so that $U_{\text{sdl-e-l}}(d') = 0$ and the exponential and linear functions have the same tangent direction at d'', as shown in Figure 4.21. This utility function is also asymptotically exponential, and we continue to use the piecewise mixture of exponential and linear functions in value iteration.

We show that the optimal policy is different from the cases of linearly and exponentially (with the same γ) soft deadline utility functions. This is the case if we let $\gamma = 0.60$,



(a) {WBBW,B}

(b) {WBB,BW}

(c) {WBB,B,W}

(d) {BW,WB,B} and {BB,BW,B}

(e) {BBB,BW}

(f) {BBB,B,W}

Figure 4.22: Optimal functional value functions of relevant states for the mixed soft deadline utility function

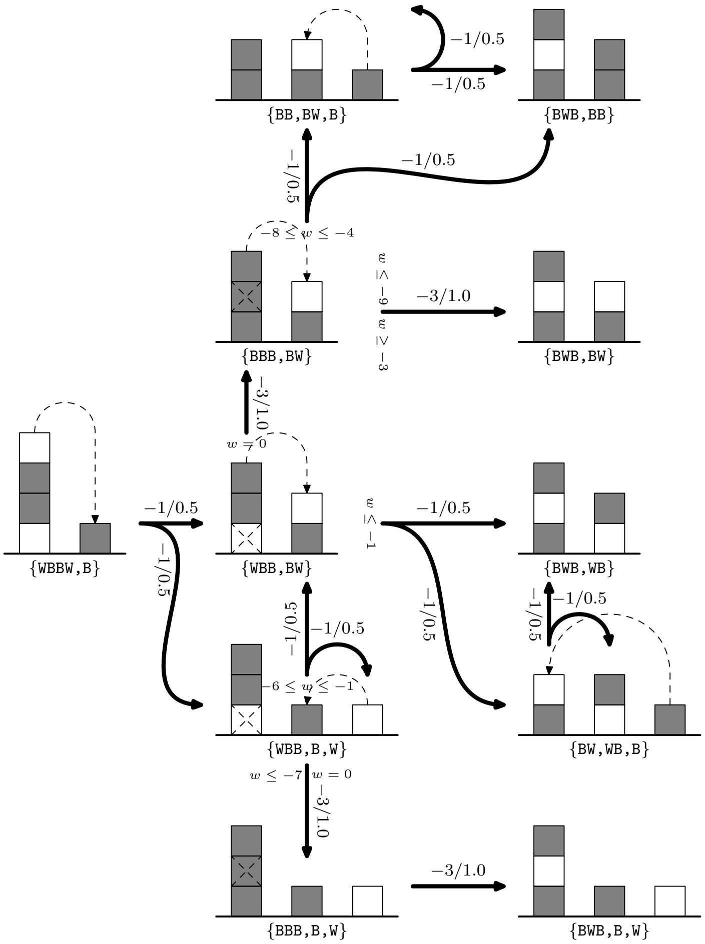


Figure 4.23: An SD-optimal policy for the augmented model with the mixed soft deadline utility function

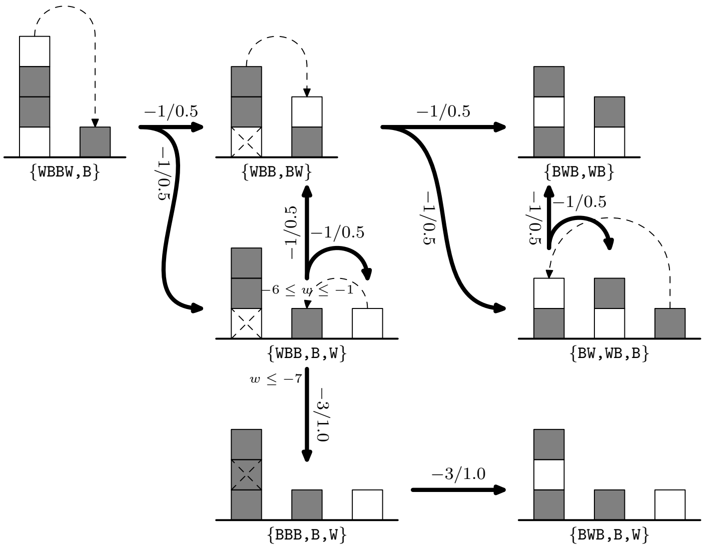


Figure 4.24: The aSD-optimal policy for the original model with the mixed soft deadline utility function

Table 4.4: Breakpoints of the value functions from Figure 4.22

{WBBW,B}	{WBB,BW}	{WBB,B,W}	{BW,WB,B} {BB,BW,B}	{BBB,BW}	{BBB,B,W}
−0.2020	−0.04584	−0.1965	−0.04584	−0.04245	−0.04245
(−8.50, −16.57)	(−9.50, −6.92)	(−6.72, −7.14)	(−9.50, −6.92)	(−8.09, −3.69)	(−7.50, −3.00)
−0.1956	−0.03820	−0.1903	−0.03820	−0.03184	0.0000
(−7.50, −10.30)	(−8.50, −4.46)	(−6.50, −6.42)	(−8.50, −4.46)	(−7.50, −2.73)	(−3.50, 1.00)
−0.1903	−0.03184	−0.1859	−0.03184	−0.02653	0.0000
(−6.50, −6.42)	(−7.50, −2.73)	(−5.72, −4.39)	(−7.50, −2.73)	(−6.50, −1.36)	(0.00, 1.00)
−0.1859	−0.02653	−0.1807	−0.02653	−0.02211	
(−5.72, −4.39)	(−6.50, −1.36)	(−5.50, −3.89)	(−6.50, −1.36)	(−5.50, −0.18)	
−0.1807	−0.02211	−0.1733	−0.02211	−0.01842	
(−5.50, −3.89)	(−5.50, −0.18)	(−4.72, −2.41)	(−5.50, −0.18)	(−4.50, 0.41)	
−0.1733	−0.01842	−0.1690	−0.01842	−0.01535	
(−4.72, −2.41)	(−4.50, 0.41)	(−4.50, −2.04)	(−4.50, 0.41)	(−3.91, 0.59)	
−0.1690	−0.01535	−0.1598	−0.01535	0.0000	
(−4.50, −2.04)	(−3.50, 0.70)	(−3.72, −1.07)	(−3.50, 0.70)	(−3.50, 1.00)	
−0.1598	−0.01279	−0.1562	−0.01279	0.0000	
(−3.72, −1.07)	(−2.50, 0.85)	(−3.50, −0.81)	(−2.50, 0.85)	(0.00, 1.00)	
−0.1562	−0.01066	−0.1460	−0.01066		
(−3.50, −0.81)	(−1.50, 0.93)	(−2.72, −0.21)	(−1.50, 0.93)		
−0.1460	−0.00888	−0.1429	−0.00888		
(−2.72, −0.21)	(−0.54, 0.96)	(−2.50, −0.05)	(−0.50, 0.96)		
−0.1429	0.0000	−0.1323	−0.00740		
(−2.50, −0.05)	(−0.50, 1.00)	(−1.72, 0.30)	(0.00, 0.97)		
−0.1323	0.0000	−0.1298			
(−1.72, 0.30)	(0.00, 1.00)	(−1.50, 0.40)			
−0.1298		−0.1191			
(−1.50, 0.40)		(−0.95, 0.55)			
−0.1191		0.0000			
(−0.72, 0.61)		(−0.50, 1.00)			
−0.1170		0.0000			
(−0.50, 0.66)		(0.00, 1.00)			
−0.1067					
(0.00, 0.74)					

$d = -6.50$, $d' = -7.50$, and $d'' = -10.50$. The functional value functions are shown in Figure 4.22. The breakpoints of the value functions are listed in Table 4.4, and we also include the coefficients of the exponential terms between breakpoints so that the piecewise mixtures can be reconstructed.

Based on the value functions, we can retrieve an optimal policy for the augmented model, as shown in Figure 4.23. The corresponding optimal policy for the original model is shown in Figure 4.24. Following the optimal policy, the agent tries the moving strategy several times before switching to the painting strategy. The resulting policy differs from the optimal policies with linearly and exponentially soft utility functions.

4.4 One-Switch Utility Functions

Since they have constant risk measures, exponential utility functions are convenient to use, but are not powerful enough to model risk attitudes that change with the wealth level. Such variable risk attitudes are realistic though. For example, if Bill Gates were faced with the lotteries given in Table 1.1 now, he would probably choose the one with the higher expected value, while he would have probably chosen the other one when he just started his company. Bell (1988) and Bell and Fishburn (2001) discussed a class of utility functions, called one-switch utility functions, which can model such changes. These functions capture the intuition that human risk attitude changes smoothly when the level of wealth is changed. An agent obeys the one-switch rule if, for every pair of lotteries for which the preference is not independent of the wealth level, there exists a threshold wealth level above which one lottery is preferred, and below which the other is preferred. In other words, as the wealth level increases, the preference between two lotteries can switch at most once. On the other hand, the linear and exponential utility functions can be called zero-switch utility functions, since they represent constant risk attitudes where the decisions do not depend on the wealth level of the decision maker.

Bell (1988) and Bell and Fishburn (2001) argued that most people are risk-averse when they have a small amount of money, and become more and more risk-neutral when they get richer and richer. If they also conform to the one-switch rule, Bell (1988) proved that the only class of utility functions satisfying these conditions is

$$U_{\text{one}}(w) = Cw - D\gamma^{w}, \qquad C, D > 0, \quad 0 < \gamma < 1. \tag{4.9}$$

Farquhar and Nakamura (1987) obtained utility functions of the same form starting with a property called the constant exchange risk property.

From the form of the utility function, we can expect that results for linear and exponential utility functions will play an important role for solving problems with one-switch utility functions. In this section, we use results from linear, exponential, and general risk-sensitive utility functions to discuss risk-sensitive planning with one-switch utility functions. Following the notation convention from Section 2.2.2.4, we use the subscript $_{\text{one}}$ to denote variables related to the MEU objective with a one-switch utility function. We also use the subscript $_{\text{one}\lll w}$ to denote variables related to "shifted" one-switch utility functions $U_{\text{one}}{\lll}w$, which we refer to as one-switch functions for later convenience (see also Section 4.3.4.3). One-switch functions are linear combinations of exponential, linear, and constant functions.

4.4.1 Finite Horizon

As we know from the results for general risk-sensitive utility functions, an optimal policy needs to be augmented-MD. In other words, we need to consider the augmented states (s, w). We therefore need to consider the (augmented) optimal values for all augmented states.

For a given policy $\langle\pi\rangle \in \langle\Pi\rangle$, we define the expected utility of the total reward over a finite horizon with initial state s as

$$\begin{aligned} V^{\langle\pi\rangle}_{\text{one},T}(s,w) &= \langle v|_{\text{one}}\rangle^{\langle\pi\rangle}_{\text{one},T}(s,w) + U_{\text{one}}(w) = v^{\Phi_w(\langle\pi\rangle)}_{\text{one}\lll w,T}(s) \\ &= E^{s,\Phi_w(\langle\pi\rangle)}\left[U_{\text{one}}(w+w_T)\right] = E^{s,\Phi_w(\langle\pi\rangle)}\left[Cw + Cw_T - D\gamma^w\gamma^{w_T}\right] \\ &= Cw + CE^{s,\Phi_w(\langle\pi\rangle)}\left[w_T\right] + D\gamma^w E^{s,\Phi_w(\langle\pi\rangle)}\left[-\gamma^{w_T}\right] \\ &= Cw + Cv^{\Phi_w(\langle\pi\rangle)}_T(s) + D\gamma^w v^{\Phi_w(\langle\pi\rangle)}_{\exp,T}(s). \end{aligned}$$

Therefore, the values under the $\mathsf{MEU}_{\text{one}}$ objective are related to the values under the MER objective and the values under the $\mathsf{MEU}_{\exp}$ objective. This relationship can be used to develop computational procedures for planning with one-switch utility functions.

We now consider the backward induction procedure for the $\mathsf{MEU}_{\text{one},T}$ objective. The basic procedure is the same as Algorithm 4.2 (BackwardInductionUtilityFunctional), but we do not need approximations, and can obtain the optimal values and an aMD-optimal policy. This is possible due to two reasons. First, we can represent the functional value functions using piecewise one-switch functions; and second, the set of piecewise one-switch functions is closed under dynamic programming operations. To understand this, we show that for all $0 \leq t \leq T$ and all states $s \in S$, the optimal functional value functions can be represented as

$$V^{*,t}_{\text{one},T}(s,w) = Cw + \text{a piecewise exponential function},$$

where each segment of the piecewise exponential function has the form $Cv + D\gamma^w v_{\exp}$ with parameters v and $v_{\exp}$. We first notice that the claim holds for T since

$$V^{*,T}_{\text{one},T}(s,w) = U_{\text{one}}(w) = Cw - D\gamma^w.$$

Suppose the claim holds for t and for all $s \in S$

$$V^{*,t}_{\text{one},T}(s,w) = Cw + V'(s,w),$$

where for each $s \in S$, $V'(s,w)$ is a piecewise exponential function. Consider epoch $t-1$. We have

$$\begin{aligned}
V^{*,t-1}_{\text{one},T}(s,w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) V^{*,t}_{\text{one},T}(s', w + r(s,a,s')) \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \big(C(w + r(s,a,s')) + V'(s', w + r(s,a,s'))\big) \\
&= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(Cw + Cr(s,a,s') + V'(s', w + r(s,a,s'))\right) \\
&= \max_{a \in A_s} \left(Cw + \sum_{s' \in S} P(s'|s,a) \big(Cr(s,a,s') + V'(s', w + r(s,a,s'))\big)\right) \\
&= Cw + \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \left(Cr(s,a,s') + V'(s', w + r(s,a,s'))\right).
\end{aligned}$$

The second term is also a piecewise exponential function since the set of piecewise exponential functions is closed under shifting, scaling, summation, and maximum operations. Therefore, the claim holds.

From the above derivation, we can see that only two values are needed for each segment (or interval on the real-line): the constant term, and the coefficient of the exponential term. The coefficient for the linear term is always C. For each state and each decision epoch, we only need to store these pairs of values along with the interval they are associated with. Therefore, we can perform the backward induction procedure with a finite number of segments since T is finite, where the operations on piecewise one-switch functions can be carried out as we discussed in Section 4.3.4.3. In fact, the operations can be simplified since the linear coefficients are always C and thus can be omitted. Especially for the maximum operation of two exponential segments, there is at most one intersection. An optimal aMD policy can be retrieved by acting greedily.

4.4.2 Infinite Horizon: Limiting Cases

If the MER values and $\mathsf{MEU}_{\text{exp}}$ values exist, then the $\mathsf{MEU}_{\text{one}}$ values also exist. We have the following relationship for these values:

$$\begin{aligned} V^{\langle\pi\rangle}_{\text{one}}(s,w) &= \lim_{T\to\infty} V^{\langle\pi\rangle}_{\text{one},T}(s,w) = \lim_{T\to\infty}\Big(Cw + Cv_T^{\Phi_w(\langle\pi\rangle)}(s) + D\gamma^w v_{\text{exp},T}^{\Phi_w(\langle\pi\rangle)}(s)\Big) \\ &= Cw + Cv^{\Phi_w(\langle\pi\rangle)}(s) + D\gamma^w v_{\text{exp}}^{\Phi_w(\langle\pi\rangle)}(s). \end{aligned} \tag{4.10}$$

We have the following result about the limiting cases for the optimal values.

Lemma 4.17. *We have*

a. $\lim\limits_{w\to\infty}\left(V^*_{\text{one}}(s,w) - Cw\right) = Cv^*(s)$;

b. $\lim\limits_{w\to-\infty} V^*_{\text{one}}(s,w)\gamma^{-w} = Dv^*_{\text{exp}}(s)$.

Proof. **a.** Let $\langle\pi\rangle^*_{\text{one}}$ be an optimal policy for the augmented model. Then

$$\begin{aligned} V^*_{\text{one}}(s,w) - Cw = V^{\langle\pi\rangle^*_{\text{one}}}_{\text{one}}(s,w) - Cw &= Cv^{\Phi_w(\langle\pi\rangle^*_{\text{one}})}(s) + D\gamma^w v_{\text{exp}}^{\Phi_w(\langle\pi\rangle^*_{\text{one}})}(s) \\ &\le Cv^*(s) + D\gamma^w v^*_{\text{exp}}(s), \end{aligned}$$

and

$$\limsup_{w\to\infty}\left(V^*_{\text{one}}(s,w) - Cw\right) \le \lim_{w\to\infty}\big(Cv^*(s) + D\gamma^w v^*_{\text{exp}}(s)\big) = Cv^*(s).$$

On the other hand, assume that π^* is an SD-optimal policy under the MER objective. We have

$$\begin{aligned} V^*_{\text{one}}(s,w) - Cw \ge V^{\Psi(\pi^*)}(s,w) - Cw &= Cv^{\Phi_w(\Psi(\pi^*))}(s) + D\gamma^w v_{\text{exp}}^{\Phi_w(\Psi(\pi^*))}(s) \\ &= Cv^{\pi^*}(s) + D\gamma^w v^{\pi^*}_{\text{exp}}(s) = Cv^*(s) + D\gamma^w v^{\pi^*}_{\text{exp}}(s), \end{aligned}$$

and

$$\liminf_{w\to\infty}\left(V^*_{\text{one}}(s,w) - Cw\right) \ge \lim_{w\to\infty}\Big(Cv^*(s) + D\gamma^w v^{\pi^*}_{\text{exp}}(s)\Big) = Cv^*(s).$$

Therefore the result holds.

b. Let $\langle\pi\rangle^*_{\text{one}}$ be an optimal policy for the augmented model. Then

$$\begin{aligned} V^*_{\text{one}}(s,w)\gamma^{-w} = V^{\langle\pi\rangle^*_{\text{one}}}_{\text{one}}(s,w)\gamma^{-w} &= Cw\gamma^{-w} + Cv^{\Phi_w(\langle\pi\rangle^*_{\text{one}})}(s)\gamma^{-w} + Dv_{\text{exp}}^{\Phi_w(\langle\pi\rangle^*_{\text{one}})}(s) \\ &\le Cw\gamma^{-w} + Cv^*(s)\gamma^{-w} + Dv^*_{\text{exp}}(s), \end{aligned}$$

and

$$\limsup_{w\to\infty} V^*_{\text{one}}(s,w)\gamma^{-w} \le \lim_{w\to\infty}\big(Cw\gamma^{-w} + Cv^*(s)\gamma^{-w} + Dv^*_{\text{exp}}(s)\big) = Dv^*_{\text{exp}}(s).$$

On the other hand, assume that π^*_{exp} is an SD-optimal policy under the $\mathsf{MEU}_{\text{exp}}$ objective. We have

$$\begin{aligned} V^*_{\text{one}}(s,w)\gamma^{-w} &\geq V^{\Psi(\pi^*_{\text{exp}})}(s,w)\gamma^{-w} = Cw\gamma^{-w} + Cv^{\Phi_w(\Psi(\pi^*_{\text{exp}}))}(s)\gamma^{-w} + Dv_{\text{exp}}^{\Phi_w(\Psi(\pi^*_{\text{exp}}))}(s) \\ &= Cw\gamma^{-w} + Cv^{\pi^*_{\text{exp}}}(s)\gamma^{-w} + Dv_{\text{exp}}^{\pi^*_{\text{exp}}}(s) = Cw\gamma^{-w} + Cv^{\pi^*_{\text{exp}}}(s)\gamma^{-w} + Dv^*_{\text{exp}}(s), \end{aligned}$$

and

$$\liminf_{w\to\infty} V^*_{\text{one}}(s,w)\gamma^{-w} \geq \lim_{w\to\infty}\Big(Cw\gamma^{-w} + Cv^{\pi^*_{\text{exp}}}(s)\gamma^{-w} + Dv^*_{\text{exp}}(s)\Big) = Dv^*_{\text{exp}}(s).$$

Therefore the result holds.

□

We next discuss properties for positive and negative models.

4.4.3 Infinite Horizon: Negative Models

For negative models, greedy policies are optimal. Define the $\mathsf{MEU}_{\text{exp}}$-optimal action set for a state $s \in S$ as

$$A^*_{\text{exp}}(s) = \arg\max_{a\in A_s} \sum_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')} v^*_{\text{exp}}(s').$$

We can determine a wealth level $\underline{w}$, below which the optimal policy can be chosen to be the same as an SD-optimal policy under the $\mathsf{MEU}_{\text{exp}}$ objective.

We first show that there exists an SD-policy π^{**}_{exp} such that its value under the MER objective satisfies

$$v^{\pi^{**}_{\text{exp}}}(s) \geq v^{\pi^*_{\text{exp}}}(s), \qquad s \in S,$$

for all π^*_{exp} such that $v_{\text{exp}}^{\pi^*_{\text{exp}}}(s) = v^*_{\text{exp}}(s)$. Consider an auxiliary MDP where the action sets are restricted to $A^*_{\text{exp}}(s)$ for each state $s \in S$. Then an SD-optimal policy under the MER objective for the auxiliary model is such a policy. Let

$$v^{**}(s) = v^{\pi^{**}_{\text{exp}}}(s), \qquad s \in S.$$

We have

$$v^{**}(s) = \max_{a\in A^*_{\text{exp}}(s)} \sum_{s'\in S} P(s'|s,a)[r(s,a,s') + v^{**}(s')], \qquad s \in S.$$

Let

$$A_{\text{exp}}^{**}(s) = \underset{a \in A_{\text{exp}}^{*}(s)}{\arg\max} \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^{**}(s')], \qquad s \in S.$$

We also define

$$q_{\text{exp}}^{*}(s,a) = \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} v_{\text{exp}}^{*}(s'), \qquad s \in S,\ a \in A_s,$$

and

$$q^{**}(s,a) = \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^{**}(s')], \qquad s \in S, a \in A_s.$$

Theorem 4.18. *Suppose Condition 2.1 (Finite Model) and Condition 2.5 (Negative Model) hold. Also assume the agent has a one-switch utility function. Let*

$$\underline{w} = \min_{s \in S} \min_{a \in A_s \setminus A_{\text{exp}}^{*}(s)} \log_\gamma \max\left(1, \frac{C}{D} \cdot \frac{q^{**}(s,a) - v^{**}(s)}{v_{\text{exp}}^{*}(s) - q_{\text{exp}}^{*}(s,a)}\right).$$

Then for $w \le \underline{w}$, it holds that $V_{\text{one}}^{*}(s,w) = V_{\text{one}}^{\Psi(\pi_{\text{exp}}^{**})}(s)$ for all states $s \in S$.

Proof. We have $v_{\text{exp}}^{*}(s) = \max_{a \in A_s} q_{\text{exp}}^{*}(s,a)$. Let

$$\epsilon' = \min_{s \in S}\left(v_{\text{exp}}^{*}(s) - \max_{a \in A_s \setminus A_{\text{exp}}^{*}(s)} q_{\text{exp}}^{*}(s,a)\right), \qquad \epsilon = \min(1, \epsilon'),$$

where we follow the convention that $\max_{a \in \varnothing} q_{\text{exp}}^{*}(s,a) = -\infty$. Therefore, $\epsilon > 0$.

Since $\lim_{w \to -\infty} V_{\text{one}}^{*}(s,w)\gamma^{-w} = Dv_{\text{exp}}^{*}(s)$ and there are only a finite number of states, there exists w_0 such that for all $w \le w_0$ and all states $s \in S$,

$$Dv_{\text{exp}}^{*}(s) - V_{\text{one}}^{*}(s,w)\gamma^{-w} \le \frac{D}{2}\epsilon. \tag{4.11}$$

Since there exists an SD-optimal policy $\langle\pi\rangle_{\text{one}}^{*}$ for the augmented model, it holds that for all states $s \in S$ and all wealth levels $w \in W$,

$$V_{\text{one}}^{*}(s,w) = V_{\text{one}}^{\langle\pi\rangle_{\text{one}}^{*}}(s,w) = Cw + Cv^{\Phi_w(\langle\pi\rangle_{\text{one}}^{*})}(s) + D\gamma^w v_{\text{exp}}^{\Phi_w(\langle\pi\rangle_{\text{one}}^{*})}(s).$$

To simplify the notation, let $\varphi_w = \Phi_w(\langle\pi\rangle_{\text{one}}^{*})$ be the aSD policy corresponding to the SD-optimal augmented policy with an initial wealth level w. The above formula can then be rewritten as

$$V_{\text{one}}^{*}(s,w) = Cw + Cv^{\varphi_w}(s) + D\gamma^w v_{\text{exp}}^{\varphi_w}(s).$$

Notice that φ_w is an HD policy in the original model. Let $\varphi_w = (d_0, d_1, \ldots, d_t, \ldots)$. We now show that if $w \leq w_0$, for all states $s \in S$, it holds that $d_0(s) \in A^*_{\text{exp}}(s)$. Suppose otherwise: there exists $s' \in S$, such that $a' = d_0(s') \notin A^*_{\text{exp}}(s')$. Then we have

$$v^{\varphi_w}_{\text{exp}}(s') = \sum_{s'' \in S} P(s''|s', a') \gamma^{r(s',a',s'')} v^{\theta(\varphi_w)}_{\text{exp}}(s'')$$

where $\theta(\varphi_w) = (d_1, d_2, \ldots, d_t, \ldots)$ is the policy obtained by shifting the decision rules of φ_w one epoch earlier. Therefore, we have

$$\begin{aligned} v^{\varphi_w}_{\text{exp}}(s') &= \sum_{s'' \in S} P(s''|s', a') \gamma^{r(s',a',s'')} v^{\theta(\varphi_w)}_{\text{exp}}(s'') \\ &\leq \sum_{s'' \in S} P(s''|s', a') \gamma^{r(s',a',s'')} v^{*}_{\text{exp}}(s'') = q^*_{\text{exp}}(s', a'). \end{aligned}$$

Since we consider negative models, it holds that $w \leq 0$ and $v^{\varphi_w}(s') \leq 0$. Therefore,

$$V^*_{\text{one}}(s', w) = Cw + Cv^{\varphi_w}(s') + D\gamma^w v^{\varphi_w}_{\text{exp}}(s') \leq D\gamma^w v^{\varphi_w}_{\text{exp}}(s') \leq D\gamma^w q^*_{\text{exp}}(s', a').$$

But in this case,

$$Dv^*_{\text{exp}}(s') - V^*_{\text{one}}(s', w)\gamma^{-w} \geq D\big(v^*_{\text{exp}}(s') - q^*_{\text{exp}}(s', a')\big) \geq D\epsilon,$$

which contradicts Eq. (4.11). Since the model is negative, we still have $w \leq w_0$ after d_0 is applied, and thus the same argument works for all d_t with $t \geq 0$. Therefore, it holds that for all $w \leq w_0$ and all states $s \in S$, $(\varphi_w)_t(s) \in A^*_{\text{exp}}(s)$.

Since the model is negative, if the current-time accumulated reward is $w \leq w_0$, any further action will result in a next-time accumulated reward of $w' \leq w \leq w_0$ and a next-time state s', in which case for the reason we just showed, an action from $A^*_{\text{exp}}(s')$ is still optimal for the $\mathsf{MEU}_{\text{one}}$ objective. In other words, it holds that for all $w \leq w_0$, all $t \in \mathbb{N}$, and all states $s \in S$, $d_t(s) \in A^*_{\text{exp}}(s)$. Or equivalently, φ_w is an $\mathsf{MEU}_{\text{exp}}$ optimal policy.

Since $v^{\varphi_w}_{\text{exp}}(s) = v^*_{\text{exp}}(s)$, we have $v^{\varphi_w}(s) \leq v^{\pi^{**}_{\text{exp}}}(s)$ by the definition of π^{**}_{exp}. Therefore,

$$\begin{aligned} V^*_{\text{one}}(s, w) &= Cw + Cv^{\varphi_w}(s) + D\gamma^w v^{\varphi_w}_{\text{exp}}(s) \\ &\leq Cw + Cv^{\pi^{**}_{\text{exp}}}(s) + D\gamma^w v^*_{\text{exp}}(s) = V^{\Psi(\pi^{**}_{\text{exp}})}_{\text{one}}(s, w). \end{aligned}$$

Thus, $V^*_{\text{one}}(s, w) = V^{\Psi(\pi^{**}_{\text{exp}})}_{\text{one}}(s, w)$ for $w \leq w_0$.

To determine $\underline{w}$, consider the leftmost part of the optimal functional value function. For $w \leq w_0$ and all $a \in A_s \setminus A^{**}_{\text{exp}}(s)$, it holds that

$$Cw + Cv^{**}(s) + D\gamma^w v^*_{\text{exp}}(s) \geq Cw + Cq^{**}(s, a) + D\gamma^w q^*_{\text{exp}}(s, a)$$

$$
\begin{aligned}
Cv^{**}(s) + D\gamma^w v^*_{\exp}(s) &\geq Cq^{**}(s,a) + D\gamma^w q^*_{\exp}(s,a) \\
D\gamma^w(v^*_{\exp}(s) - q^*_{\exp}(s,a)) &\geq C(q^{**}(s,a) - v^{**}(s)).
\end{aligned}
$$

As w increases, the left-hand side decreases and the right-hand side is a constant and eventually the left-hand side can be less than the right-hand side. Therefore, a candidate for the wealth level $\underline{w}$ is the switching point that the direction of the above inequality is reversed.

For all actions $a \in A_s$, we have $v^*_{\exp}(s) \geq q^*_{\exp}(s,a)$ by definition. Therefore, for an action $a \in A^*_{\exp}(s) \setminus A^{**}_{\exp}(s)$, such a reversion is impossible since $v^{**}(s) > q^{**}(s,a)$ and $v^*_{\exp}(s) = q^*_{\exp}(s,a)$. Now we consider an action $a \in A_s \setminus A^*_{\exp}(s)$. If $v^{**}(s) \geq q^{**}(s,a)$, it is impossible to choose the action a. If $v^*_{\exp}(s) > q^*_{\exp}(s,a)$ and $v^{**}(s) < q^{**}(s,a)$, then we choose a if

$$
\begin{aligned}
Cw + Cv^{**}(s) + D\gamma^w v^*_{\exp}(s) &< Cw + Cq^{**}(s,a) + D\gamma^w q^*_{\exp}(s,a) \\
D\gamma^w \left(v^*_{\exp}(s) - q^*_{\exp}(s,a)\right) &< C\left(q^{**}(s,a) - v^{**}(s)\right) \\
\gamma^w &< \frac{C}{D} \cdot \frac{q^{**}(s,a) - v^{**}(s)}{v^*_{\exp}(s) - q^*_{\exp}(s,a)} \\
w &> \log_\gamma \left(\frac{C}{D} \cdot \frac{q^{**}(s,a) - v^{**}(s)}{v^*_{\exp}(s) - q^*_{\exp}(s,a)} \right).
\end{aligned}
$$

Therefore, a candidate for $\underline{w}$ due to state s is

$$
\min_{a \in A_s \setminus A^*_{\exp}(s)} \log_\gamma \max \left(1, \frac{C}{D} \cdot \frac{q^{**}(s,a) - v^{**}(s)}{v^*_{\exp}(s) - q^*_{\exp}(s,a)} \right).
$$

We included 1 to make sure this value is nonpositive. Overall, $\underline{w}$ is chosen so that such a switching cannot happen for all states $s \in S$ if $w < \underline{w}$, and there is at least one state for which the switching happens if $w > \underline{w}$. □

We have shown that when $w(< 0)$ is sufficiently large in magnitude, an optimal policy chooses actions from $A^{**}_{\exp}(s)$ for all states $s \in S$. Since the model is negative, there are only a finite number of different $w \in [\underline{w}, 0]$, and we only need to consider a finite number of aSD policies.

4.4.3.1 Value Iteration

We can continue to the use the functional version of value iteration to solve problems under the $\mathsf{MEU}_{\text{one}}$ objective. If we start with a one-switch utility function, the functional value functions $V^t_{\text{one}}(s, \cdot)$ obtained in each step are piecewise one-switch, since the dynamic programming operations convert piecewise one-switch functions to piecewise one-switch functions. Therefore, we can represent the functional value functions with a finite representation, where three parameters are used for each segment of the piecewise one-switch value function.

4.4.3.2 Backward Induction

Since the model is negative and the optimal functional value functions are piecewise one-switch, we can use a procedure similar to backward induction to retrieve an optimal policy. The procedure is shown in Algorithm 4.4 (BackwardInductionOneSwitchNegative).

Since we know that the optimal functional value function is piecewise one-switch for each state, we can represent the value function with a list of tuples $(w^i(s), v^i(s), v^i_{\text{exp}}(s), a^i(s))$ ordered in w^i for $i = 1, 2, \ldots, n$. We refer to this list as *VAList*, which is accessed using the procedure GetValues. GetValues(VAList, s, w) retrieves the i-th tuple from the list for state s such that $w^i(s) < w \leq w^{i+1}(s)$. We maintain the following property in the backward induction procedure: for each $w \in (w^i(s), w^{i+1}(s)]$, we have $V^*_{\text{one}}(s, w) = Cw + Cv^i(s) + D\gamma^w v^i_{\text{exp}}(s)$, and the action to perform is $a^i(s)$.

According to Theorem 4.18, for each state s, we can set $w^0(s) = -\infty$, $v^0(s) = v^{**}(s)$, $v^0_{\text{exp}}(s) = v^*_{\text{exp}}(s)$, and $a^0(s) = \pi^{**}_{\text{exp}}(s)$. They represent the values and policy for $w \leq \underline{w}$. Lines 1–4 perform this initialization.

Lines 5–14 calculates $\underline{w}$ according to Theorem 4.18. At the same time, the algorithm also initializes the priority queue PQ to be used later. These lines take care of the leftmost infinite intervals of the optimal value function.

Consider a state s and an interval $(w^i(s), w^i(s) + \delta]$ for a small $\delta > 0$. Assume that for all $w \in (w^i(s), w^i(s) + \delta]$, all actions $a \in A_s$, and all states $s' \in \text{succ}(s, a)$, there exists $j_{s'}$ such that $w + r(s, a, s') \in (w^{j_{s'}}(s'), w^{j_{s'}+1}(s')]$ and the related optimal values and actions are determined. We need to determine the values $v^i(s)$ and $v^i_{\text{exp}}(s)$ for the interval $(w^i(s), w^i(s) + \delta]$. This can be done by checking the optimality equations for s and $w \in (w^i(s), w^i(s) + \delta]$. The optimality equation is

$$V^*_{\text{one}}(s, w) = \max_{a \in A_s} Q^*_{\text{one}}(s, w, a)$$

Algorithm 4.4 Backward Induction for Negative Models under the $\mathsf{MEU}_{\text{one}}$ Objective

$\mathit{VAList} = \mathsf{BackwardInductionOneSwitchNegative}(M, \gamma, C, D)$

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- γ, C, D, parameters of the one-switch utility function, $0 < \gamma < 1, C > 0, D > 0$;

Output:

- *VAList*, the list representing the optimal value function and an optimal policy;

Local:

- *PQ*, a priority queue;

```
determine $v^*_{\text{exp}}$, $v^{**}$, and $\pi^{**}_{\text{exp}}$;
for all $s \in S$ do
    AddList($\mathit{VAList}, s, -\infty, v^{**}(s), v^*_{\text{exp}}(s), \pi^{**}_{\text{exp}}(s)$);
end for
for all $s \in S$ do
    for all $a \in A_s$ do
        $q^i(s,a), q^i_{\text{exp}}(s,a) \leftarrow 0$;
        for all $s' \in \text{succ}(s,a)$ do
            $q^i(s,a) \leftarrow q^i(s,a) + P(s'|s,a)[r(s,a,s') + v^{**}(s')]$;
            $q^i_{\text{exp}}(s,a) \leftarrow q^i_{\text{exp}}(s,a) + P(s'|s,a)\gamma^{r(s,a,s')}v^*_{\text{exp}}(s')$;
        end for
        InsertSwitchingPointNegative($PQ, s, -\infty, v^{**}(s), v^*_{\text{exp}}(s), q^i(s,a), q^i_{\text{exp}}(s,a)$);
    end for
end for
repeat
    $s, w^i \leftarrow$ ExtractMin($PQ$);
    $V^i_{\text{one}} \leftarrow -\infty$;
    for all $a \in A_s$ do
        $q^i(s,a), q^i_{\text{exp}}(s,a) \leftarrow 0$;
        for all $s' \in \text{succ}(s,a)$ do
            $v^j(s'), v^j_{\text{exp}}(s') \leftarrow$ GetValues($\mathit{VAList}, s', w^i + r(s,a,s')$);
            $q^i(s,a) \leftarrow q^i(s,a) + P(s'|s,a)[r(s,a,s') + v^j(s')]$;
            $q^i_{\text{exp}}(s,a) \leftarrow q^i_{\text{exp}}(s,a) + P(s'|s,a)\gamma^{r(s,a,s')}v^j_{\text{exp}}(s')$;
        end for
        $Q^i_{\text{one}}(s,a) \leftarrow Cw^i + Cq^i(s,a) + D\gamma^{w^i}q^i_{\text{exp}}(s,a)$;
        if $Q^i_{\text{one}}(s,a) > V^i_{\text{one}}$ or ($Q^i_{\text{one}}(s,a) = V^i_{\text{one}}$ and $q^i_{\text{exp}}(s,a) < v^i_{\text{exp}}$) then
            $V^i_{\text{one}} \leftarrow Q^i_{\text{one}}(s,a)$;
            $a^i, v^i, v^i_{\text{exp}} \leftarrow a, q^i(s,a), q^i_{\text{exp}}(s,a)$;
        end if
    end for
    AddList($\mathit{VAList}, s, w^i, v^i, v^i_{\text{exp}}, a^i$);
    for all $a \in A_s$ do
        InsertSwitchingPointNegative($PQ, s, w^i, v^i, v^i_{\text{exp}}, q^i(s,a), q^i_{\text{exp}}(s,a)$);
    end for
    for all $a \in A$ do
        for all $s' \in \text{pred}(s,a)$ do
            InsertNegative($PQ, s', w^i - r(s,a,s')$);
        end for
    end for
until IsEmpty($PQ$);
```

Algorithm 4.5 Calculate and Insert the Switching Point for Negative Models

```
InsertSwitchingPointNegative($PQ, s, w, v, v_{\exp}, q, q_{\exp}$)
if $v_{\exp} \neq q_{\exp}$ then
    $temp \leftarrow \frac{q - v}{v_{\exp} - q_{\exp}}$;
    if $temp > 0$ then
        $\hat{w} \leftarrow \log_\gamma \left(\frac{C}{D} \cdot temp\right)$;
        if $\hat{w} > w$ then
            InsertNegative($PQ, s, \hat{w}$);
        end if
    end if
end if
```

where

$$\begin{aligned}
Q^*_{\text{one}}(s, w, a) &= \sum_{s' \in S} P(s'|s,a) V^*_{\text{one}}(s', w + r(s,a,s')) \\
&= \sum_{s' \in S} P(s'|s,a) \Big(C(w + r(s,a,s')) + C v^{j_{s'}}(s') + D\gamma^{w + r(s,a,s')} v^{j_{s'}}_{\exp}(s') \Big) \\
&= Cw + C \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^{j_{s'}}(s')] + D\gamma^w \sum_{s' \in S} P(s'|s,a) \gamma^{r(s,a,s')} v^{j_{s'}}_{\exp}(s').
\end{aligned}$$

Since the utility function and thus the value functions are continuous in w, we can determine the best action a^i for this small interval using the value $w^i(s)$ wherever w appears. Then we have

$$v^i(s) = \sum_{s' \in S} P(s'|s,a^i)[r(s,a^i,s') + v^{j_{s'}}(s')]$$

and

$$v^i_{\exp}(s) = \sum_{s' \in S} P(s'|s,a^i) \gamma^{r(s,a^i,s')} v^{j_{s'}}_{\exp}(s').$$

These calculations are done on Lines 17–31.

Now we consider how far to the right the interval $(w^i(s), w^i(s) + \delta]$ can extend, and thus determine $w^{i+1}(s)$. It is sufficient to consider the switching point $\hat{w}$ to prefer an action $a \neq a^i$ over a^i, provided that for all s' and all $w \in (w^i(s), \hat{w}]$, it holds that $w + r(s,a',s') \in (w^{j_{s'}}(s'), w^{j_{s'}+1}(s')]$. Similar to the proof of Theorem 4.18, the action a^i is preferred over another action $a \neq a^i$ as long as

$$Cw + Cv^i(s) + D\gamma^w v^i_{\exp}(s) \geq Cw + Cq^i(s,a) + D\gamma^w q^i_{\exp}(s,a)$$

$$Cv^i(s) + D\gamma^w v^i_{\exp}(s) \geq Cq^i(s,a) + D\gamma^w q^i_{\exp}(s,a).$$

Algorithm 4.6 Customized Insertion Operation for Algorithm 4.4

```
InsertNegative($PQ, s, w$)
if $w < 0$ then
    if IsMember($PQ, s$) then
        if GetKey($PQ, s$) $> w$ then
            DecreaseKey($PQ, s, w$);
        end if
    else
        Insert($PQ, s, w$);
    end if
end if
```

where

$$q^i(s,a) = \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^{j_{s'}}(s')]$$

and

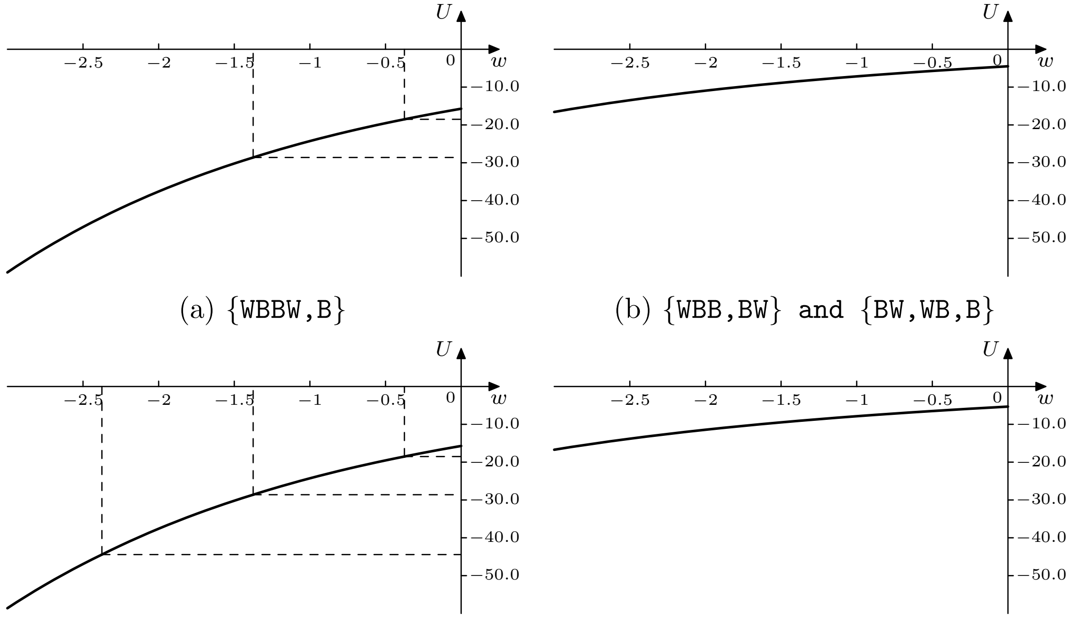
$$q^i_{\exp}(s,a) = \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} v^{j_{s'}}_{\exp}(s').$$

Since a^i is optimal for w close to $w^i(s)$, it is impossible to have $v^i(s) < q^i(s,a)$ and $v^i_{\exp}(s) < q^i_{\exp}(s,a)$. If $v^i(s) \geq q^i(s,a)$ and $v^i_{\exp}(s) \geq q^i_{\exp}(s,a)$, then a^i is always preferred over a. Otherwise, the switching point to prefer a over a^i is

$$\hat{w} = \log_\gamma \left(\frac{C}{D} \cdot \frac{q^i(s,a) - v^i(s)}{v^i_{\exp}(s) - q^i_{\exp}(s,a)} \right).$$

Then the switching point to prefer an action other than a^i can be determined by considering all actions in A_s, and it is also the value for $w^{i+1}(s)$. The switching points are calculated on Lines 32–34. We actually determine all potential switching points where another action is preferred over the current one, but the closest one will be determined by the priority queue PQ (see below). Notice that we still need to make sure that $w + r(s,a,s') \in (w^{j_{s'}}(s'), w^{j_{s'}+1}(s')]$ for all $w \in (w^i(s), \hat{w}]$. We guarantee this by adding potential switching points for those w values that lead to known switching points, that is, w values such that $w + r(s,a,s')$ is a switching point for some states s, s' and action a. These switching points are taken into account on Lines 35–39.

Since we need to determine the optimal values and actions for $(w^{j_{s'}}(s'), w^{j_{s'}+1}(s')]$ before we determine those for $(w^i(s), w^{i+1}(s)]$, we need to proceed in the order of increasing w. Therefore, we use a priority queue PQ that stores pairs (s, w) where w is used as the key,



(a) {WBBW,B}

(b) {WBB,BW} and {BW,WB,B}

(c) {WBB,B,W}

(d) {BBB,B,W}

Figure 4.25: Optimal functional value functions of relevant states for the one-switch utility function

Table 4.5: Segments of the value functions from Figure 4.25

						{WBB,B,W}					
{WBBW,B}			{WBB,BW}			{BW,WB,B}			{BBB,BW}		
−22.03	−5.00		−5.00	−2.00		−21.43	−6.00		−4.63	−3.00	
		(−1.38, −28.61)			(0.00, −4.50)			(−2.38, −44.43)			(0.00, −5.31)
−22.52	−4.50					−22.03	−5.00				
		(−0.38, −18.52)						(−1.38, −28.61)			
−22.94	−4.25					−22.52	−4.50				
		(0.00, −15.72)						(−0.38, −18.52)			
						−22.94	−4.25				
								(0.00, −15.72)			

and a smaller value has a higher priority. The InsertNegative operation is a customized version of a regular insertion operation for priority queues and shown in Algorithm 4.6 (InsertNegative). The ExtractMin operation removes the entry with the smallest key and returns it. The backward induction procedure terminates when the queue is empty.

Using the algorithm, we can obtain an SD-optimal policy for the augmented model for the painted-blocks problem in Figure 4.6. The policy is also aSD-optimal for the original model, as shown in Figure 4.26. We also obtain the value functions for the relevant states, as shown in Figure 4.25. The v_{exp} and v values of the segments of the value functions are shown in Table 4.5, together with the switching points, where the first columns are v_{exp} values, the second columns are v values, and the switching points are enclosed in parenthesis.

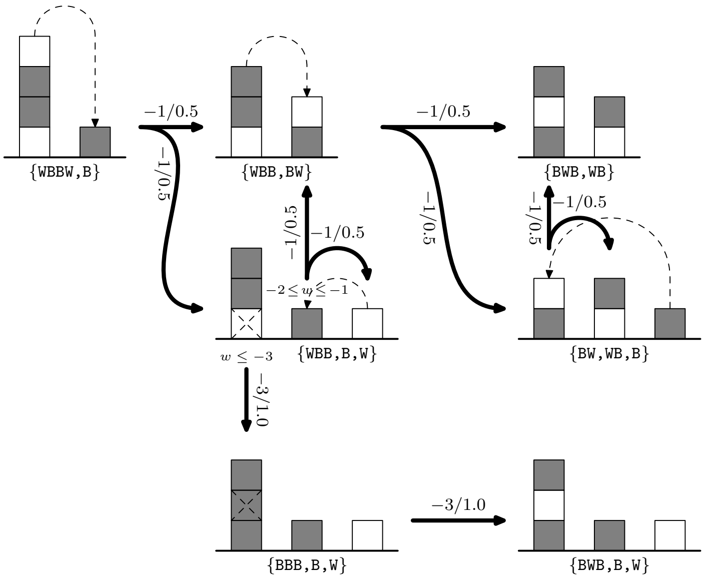


Figure 4.26: An aSD-optimal policy for the one-switch utility function

4.4.4 Infinite Horizon: Positive Models

Define the MER-optimal action set for a state $s \in S$ as

$$A^*(s) = \arg\max_{a \in A_s} P(s'|s,a)[r(s,a,s') + v^*(s')].$$

We can determine a wealth level $\overline{w}$, above which the optimal policy can be chosen to be the same as an SD-optimal under the MER objective.

We first show that there exists an SD-policy π^{**} such that its value under the $\mathsf{MEU}_{\text{exp}}$ objective satisfies

$$v_{\text{exp}}^{\pi^{**}}(s) \geq v_{\text{exp}}^{\pi^*}(s), \qquad s \in S,$$

for all π^* such that $v^{\pi^*}(s) = v^*(s)$. Consider an auxiliary MDP where the action sets are restricted to $A^*(s)$ for each state $s \in S$. Then an SD-optimal policy under the $\mathsf{MEU}_{\text{exp}}$ objective for the auxiliary model is such a policy. Let

$$v_{\text{exp}}^{**}(s) = v_{\text{exp}}^{\pi^{**}}(s), \qquad s \in S.$$

We have

$$v_{\text{exp}}^{**}(s) = \max_{a \in A^*(s)} P(s'|s,a)\gamma^{r(s,a,s')} v_{\text{exp}}^{**}(s'), \qquad s \in S.$$

Let

$$A^{**}(s) = \arg\max_{a \in A^*(s)} P(s'|s,a)\gamma^{r(s,a,s')} v_{\text{exp}}^{**}(s'), \qquad s \in S.$$

We also define

$$q^*(s,a) = \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^*(s')], \qquad s \in S,\ a \in A_s.$$

and

$$q_{\text{exp}}^{**}(s,a) = \sum_{s' \in S} P(s'|s,a)\gamma^{r(s,a,s')} v_{\text{exp}}^{**}(s'), \qquad s \in S, a \in A_s.$$

Theorem 4.19. *Suppose Condition 2.1 (Finite Model) and Condition 2.7 (Positive Model) hold. Also assume the agent has a one-switch utility function. Let*

$$\overline{w} = \max_{s \in S} \max_{a \in A_s \setminus A^*(s)} \log_\gamma \max\left(1, \frac{C}{D} \cdot \frac{v^*(s) - q^*(s,a)}{q_{\text{exp}}^{**}(s,a) - v_{\text{exp}}^{**}(s)}\right).$$

Then for $w \geq \overline{w}$, it holds that $V_{\text{one}}^*(s,w) = V_{\text{one}}^{\Psi(\pi^{**})}(s)$ for all states $s \in S$.

Proof. We have $v^*(s) = \max_{a \in A_s} q^*(s,a)$. Let

$$\epsilon' = \min_{s \in S}\left(v^*(s) - \max_{a \in A_s \setminus A^*(s)} q^*(s,a)\right), \qquad \epsilon = \min(1, \epsilon'),$$

where we follow the convention that $\max_{a \in \varnothing} q^*(s,a) = -\infty$. Therefore, $\epsilon > 0$.

Since $\lim_{w \to \infty} V_{\text{one}}^*(s,w) - Cw = Cv^*(s)$ and there are only a finite number of states, there exists w_0 such that for all $w \geq w_0$ and all states $s \in S$,

$$Cw + Cv^*(s) - V_{\text{one}}^*(s,w) \leq \frac{C}{2}\epsilon. \tag{4.12}$$

Since there exists an SD-optimal policy $\langle\pi\rangle_{\text{one}}^*$ for the augmented model, it holds that for all states $s \in S$ and all wealth levels $w \in W$,

$$V_{\text{one}}^*(s,w) = V_{\text{one}}^{\langle\pi\rangle_{\text{one}}^*}(s,w) = Cw + Cv^{\Phi_w(\langle\pi\rangle_{\text{one}}^*)}(s) + D\gamma^w v_{\text{exp}}^{\Phi_w(\langle\pi\rangle_{\text{one}}^*)}(s).$$

To simplify the notation, let $\varphi_w = \Phi_w(\langle\pi\rangle_{\text{one}}^*)$ be the aSD policy corresponding to the SD-optimal augmented policy with an initial wealth level w. The above formula can then be rewritten as

$$V_{\text{one}}^*(s,w) = Cw + Cv^{\varphi_w}(s) + D\gamma^w v_{\text{exp}}^{\varphi_w}(s).$$

Notice that φ_w is an HD policy in the original model. Let $\varphi_w = (d_0, d_1, \ldots, d_t, \ldots)$. We now show when $w \geq w_0$, for all states $s \in S$, it holds that $d_0(s) \in A^*(s)$. Suppose otherwise: there exists $s' \in S$, such that $a' = (\varphi_w)_0(s') \notin A^*(s')$. Then we have

$$v^{\varphi_w}(s') = \sum_{s'' \in S} P(s''|s',a')[r(s',a',s'') + v^{\theta(\varphi_w)}(s'')]$$

where $\theta(\varphi_w) = (d_1, d_2, \ldots, d_t, \ldots)$ is the policy obtained by shifting the decision rules of φ_w one epoch earlier. Therefore, we have

$$\begin{aligned} v^{\varphi_w}(s') &= \sum_{s'' \in S} P(s''|s', a')[r(s', a', s'') + v^{\theta(\varphi_w)}(s'')] \\ &\leq \sum_{s'' \in S} P(s''|s', a')[r(s', a', s'') + v^*(s'')] = q^*(s', a'). \end{aligned}$$

Since $0 < \gamma < 1$, it holds that $v_{\text{exp}}^{\varphi_w}(s') \leq 0$. Therefore

$$V_{\text{one}}^*(s', w) = Cw + Cv^{\varphi_w}(s') + D\gamma^w v_{\text{exp}}^{\varphi_w}(s') \leq Cw + Cv^{\varphi_w}(s') \leq Cw + Cq^*(s', a').$$

But in this case,

$$Cw + Cv^*(s') - V_{\text{one}}^*(s', w) \geq C\big(v^*(s') - q^*(s', a')\big) \geq C\epsilon,$$

which contradicts Eq. (4.12). Since the model is positive, we still have $w \geq w_0$ after d_0 is applied, and thus the same argument works for all d_t with $t \geq 0$. Therefore, it holds that for all $w \geq w_0$ and all states $s \in S$, $(\varphi_w)_t(s) \in A^*(s)$.

Since the model is positive, if the current-time accumulated reward is $w \geq w_0$, any further action will result in a next-time accumulated reward of $w' \geq w \geq w_0$ and a next-time state s', in which case for the reason we just showed, an action from $A^*(s')$ is still optimal for the $\mathsf{MEU}_{\text{one}}$ objective. In other words, it holds that for all $w \geq w_0$, all $t \in \mathbb{N}$, and all states $s \in S$, $d_t(s) \in A^*(s)$. Or equivalently, φ_w is an $\mathsf{MEU}_{\text{exp}}$ optimal policy.

Since $v^{\varphi_w}(s) = v^*(s)$, we have $v_{\text{exp}}^{\varphi_w}(s) \leq v_{\text{exp}}^{\pi^{**}}(s)$ by the definition of π^{**}. Therefore,

$$\begin{aligned} V_{\text{one}}^*(s, w) &= Cw + Cv^{\varphi_w}(s) + D\gamma^w v_{\text{exp}}^{\varphi_w}(s) \\ &\leq Cw + Cv^*(s) + D\gamma^w v_{\text{exp}}^{\pi^{**}}(s) = V_{\text{one}}^{\Psi(\pi^{**})}(s, w). \end{aligned}$$

Thus, $V_{\text{one}}^*(s, w) = V_{\text{one}}^{\Psi(\pi^{**})}(s, w)$ for $w \geq w_0$.

To determine $\overline{w}$, consider the rightmost part of the optimal functional value function, for which we must have for all $a \in A_s \setminus A^{**}(s)$, it holds that

$$Cw + Cv^*(s) + D\gamma^w v_{\text{exp}}^{**}(s) \geq Cw + Cq^*(s, a) + D\gamma^w q_{\text{exp}}^{**}(s, a)$$

$$Cv^*(s) + D\gamma^w v_{\text{exp}}^{**}(s) \geq Cq^*(s, a) + D\gamma^w q_{\text{exp}}^{**}(s, a)$$

$$C(v^*(s) - q^*(s, a)) \geq D\gamma^w (q_{\text{exp}}^{**}(s, a) - v_{\text{exp}}^{**}(s)).$$

As w decreases, the right-hand side increases and the left-hand side is a constant and eventually the right-hand side can be greater than the left-hand side. Therefore, a candidate for the wealth level $\overline{w}$ is the switching point that the direction of the above inequality is reversed.

For all actions $a \in A_s$, we have $v^*(s) \geq q^*(s,a)$ by definition. Therefore, for an action $a \in A^*(s) \setminus A^{**}(s)$, such a reversion is impossible since $v^{**}_{\text{exp}}(s) > q^{**}_{\text{exp}}(s,a)$ and $v^*(s) = q^*(s,a)$. Now we consider an action $a \in A_s \setminus A^*(s)$. If $v^{**}_{\text{exp}}(s) \geq q^{**}_{\text{exp}}(s,a)$, it is impossible to choose the action a. If $v^*(s) > q^*(s,a)$ and $v^{**}_{\text{exp}}(s) < q^{**}_{\text{exp}}(s,a)$, then we choose a if

$$\begin{aligned}
Cw + Cv^*(s) + D\gamma^w v^{**}_{\text{exp}}(s) &< Cw + Cq^*(s,a) + D\gamma^w q^{**}_{\text{exp}}(s,a) \\
C\big(v^*(s) - q^*(s,a)\big) &< D\gamma^w \left(q^{**}_{\text{exp}}(s,a) - v^{**}_{\text{exp}}(s)\right) \\
\gamma^w &> \frac{C}{D}\frac{v^*(s) - q^*(s,a)}{q^{**}_{\text{exp}}(s,a) - v^{**}_{\text{exp}}(s)} \\
w &< \log_\gamma\left(\frac{C}{D}\frac{v^*(s) - q^*(s,a)}{q^{**}_{\text{exp}}(s,a) - v^{**}_{\text{exp}}(s)}\right).
\end{aligned}$$

Therefore, a candidate for $\overline{w}$ due to the state s is

$$\max_{a \in A_s \setminus A^*_{\text{exp}}(s)} \log_\gamma \min\left(1, \frac{C}{D} \cdot \frac{q^{**}(s,a) - v^{**}(s)}{v^*_{\text{exp}}(s) - q^*_{\text{exp}}(s,a)}\right).$$

We included 1 to make sure this value is nonnegative. Overall, $\overline{w}$ is chosen so that such a switching cannot happen for all states $s \in S$ if $w > \overline{w}$, and there is at least one state for which the switching happens if $w < \overline{w}$. □

We have shown that when w is sufficiently large, an optimal policy chooses actions from $A^{**}(s)$ for all states $s \in S$. Since there are only a finite number of different $w \in [0, \overline{w}]$, we only need to consider a finite number of aSD policies.

4.4.4.1 Value Iteration

We can also use value iteration to solve MDPs under the $\mathsf{MEU}_{\text{one}}$ objective, and the algorithm is exactly the same as that for negative models.

4.4.4.2 Backward Induction

Since the model is positive and the optimal functional value functions are piecewise one-switch, we can use a procedure similar to backward induction to retrieve an optimal policy. The procedure is shown in Algorithm 4.7 (BackwardInductionOneSwitchPositive).

Since we know that the optimal functional value function is piecewise one-switch for each state, we can represent the value function with a list of tuples $(w^i(s), v^i(s), v^i_{\text{exp}}(s), a^i(s))$ ordered in w^i for $i = 1, 2, \ldots, n$. We refer to this list as *VAList*, which is accessed using the procedure GetValues. GetValues(VAList, s, w) retrieves the i-th tuple from the list for state

s such that $w^{i+1}(s) \leq w < w^i(s)$. We maintain the following property in the backward induction procedure: for each $w \in [w^{i+1}(s), w^i(s))$, we have $V^*_{\text{one}}(s,w) = Cw + Cv^i(s) + D\gamma^w v^i_{\text{exp}}(s)$, and the action to perform is $a^i(s)$.

According to Theorem 4.19, for each state s, we can set $w^0(s) = \infty$, $v^0(s) = v^*(s)$, $v^0_{\text{exp}}(s) = v^{**}_{\text{exp}}(s)$, and $a^0(s) = \pi^{**}(s)$. They represent the values and policy for $w \geq \overline{w}$. Lines 1–4 perform this initialization.

Lines 5–14 calculates $\overline{w}$ according to Theorem 4.19. At the same time, the algorithm also initializes the priority queue PQ to be used later. These lines take care of the leftmost infinite intervals of the optimal value function.

Consider a state s and an interval $[w^i(s) - \delta, w^i(s))$ for a small $\delta > 0$. Assume that for all $w \in [w^i(s) - \delta, w^i(s))$, all actions $a \in A_s$, and all states $s' \in \text{succ}(s,a)$, there exists $j_{s'}$ such that $w + r(s,a,s') \in [w^{j_{s'}+1}(s'), w^{j_{s'}+1}(s'))$ and the related optimal values and actions are determined. We need to determine the values $v^i(s)$ and $v^i_{\text{exp}}(s)$ for the interval $[w^i(s) - \delta, w^i(s))$. This can be done by checking the optimality equations for s and $w \in [w^i(s) - \delta, w^i(s))$. The optimality equation is

$$V^*_{\text{one}}(s,w) = \max_{a \in A_s} Q^*_{\text{one}}(s,w,a)$$

where

$$\begin{aligned}
Q^*_{\text{one}}(s,w,a) &= \sum_{s' \in S} P(s'|s,a) V^*_{\text{one}}(s', w + r(s,a,s')) \\
&= \sum_{s' \in S} P(s'|s,a) \Big(C(w + r(s,a,s')) + Cv^{j_{s'}}(s') + D\gamma^{w + r(s,a,s')} v^{j_{s'}}_{\text{exp}}(s') \Big) \\
&= Cw + C \sum_{s' \in S} P(s'|s,a)[r(s,a,s') + v^{j_{s'}}(s')] + D\gamma^w \sum_{s' \in S} P(s'|s,a) \gamma^{r(s,a,s')} v^{j_{s'}}_{\text{exp}}(s').
\end{aligned}$$

Since the utility function and thus the value functions are continuous in w, we can determine the best action a^i for this small interval using the value $w^i(s)$ wherever w appears. Then we have

$$v^i(s) = \sum_{s' \in S} P(s'|s,a^i)[r(s,a^i,s') + v^{j_{s'}}(s')]$$

and

$$v^i_{\text{exp}}(s) = \sum_{s' \in S} P(s'|s,a^i) \gamma^{r(s,a^i,s')} v^{j_{s'}}_{\text{exp}}(s').$$

Algorithm 4.7 Backward Induction for Positive Models under the $\mathsf{MEU}_{\text{one}}$ Objective

$\mathit{VAList} = \mathsf{BackwardInductionOneSwitchPositive}(M, \gamma, C, D)$

Input:

- $M = (S, A, P, r)$, a finite MDP model;
- γ, C, D, parameters of the one-switch utility function, $0 < \gamma < 1, C > 0, D > 0$;

Output:

- *VAList*, the list representing the optimal value function and an optimal policy;

Local:

- *PQ*, a priority queue;

```
determine $v^*$, $v^{**}_{\text{exp}}$, and $\pi^{**}$;
for all $s \in S$ do
    AddList($\mathit{VAList}, s, \infty, v^*(s), v^{**}_{\text{exp}}(s), \pi^{**}(s)$);
end for
for all $s \in S$ do
    for all $a \in A_s$ do
        $q^i(s,a), q^i_{\text{exp}}(s,a) \leftarrow 0$;
        for all $s' \in \text{succ}(s,a)$ do
            $q^i(s,a) \leftarrow q^i(s,a) + P(s'|s,a)[r(s,a,s') + v^*(s')]$;
            $q^i_{\text{exp}}(s,a) \leftarrow q^i_{\text{exp}}(s,a) + P(s'|s,a)\gamma^{r(s,a,s')} v^{**}_{\text{exp}}(s')$;
        end for
        InsertSwitchingPointPositive($PQ, s, -\infty, v^*(s), v^{**}_{\text{exp}}(s), q^i(s,a), q^i_{\text{exp}}(s,a)$);
    end for
end for
repeat
    $s, w^i \leftarrow$ ExtractMax($PQ$);
    $V^i_{\text{one}} \leftarrow \infty$;
    for all $a \in A_s$ do
        $q^i(s,a), q^i_{\text{exp}}(s,a) \leftarrow 0$;
        for all $s' \in \text{succ}(s,a)$ do
            $v^j(s'), v^j_{\text{exp}}(s') \leftarrow$ GetValues($\mathit{VAList}, s', w^i + r(s,a,s')$);
            $q^i(s,a) \leftarrow q^i(s,a) + P(s'|s,a)[r(s,a,s') + v^j(s')]$;
            $q^i_{\text{exp}}(s,a) \leftarrow q^i_{\text{exp}}(s,a) + P(s'|s,a)\gamma^{r(s,a,s')} v^j_{\text{exp}}(s')$;
        end for
        $Q^i_{\text{one}}(s,a) \leftarrow Cw^i + Cq^i(s,a) + D\gamma^{w^i} q^i_{\text{exp}}(s,a)$;
        if $Q^i_{\text{one}}(s,a) > V^i_{\text{one}}$ or ($Q^i_{\text{one}}(s,a) = V^i_{\text{one}}$ and $q^i_{\text{exp}}(s,a) < v^i_{\text{exp}}$) then
            $V^i_{\text{one}} \leftarrow Q^i_{\text{one}}(s,a)$;
            $a^i, v^i, v^i_{\text{exp}} \leftarrow a, q^i(s,a), q^i_{\text{exp}}(s,a)$;
        end if
    end for
    AddList($\mathit{VAList}, s, w^i, v^i, v^i_{\text{exp}}, a^i$);
    for all $a \in A_s$ do
        InsertSwitchingPointPositive($PQ, s, w^i, v^i, v^i_{\text{exp}}, q^i(s,a), q^i_{\text{exp}}(s,a)$);
    end for
    for all $a \in A$ do
        for all $s' \in \text{pred}(s,a)$ do
            InsertPositive($PQ, s', w^i - r(s,a,s')$);
        end for
    end for
until IsEmpty($PQ$);
```

Algorithm 4.8 Calculate and Insert the Switching Point for Positive Models

```
InsertSwitchingPointPositive($PQ, s, w, v, v_{\exp}, q, q_{\exp}$)
if $v_{\exp} \neq q_{\exp}$ then
    $temp \leftarrow \frac{q - v}{v_{\exp} - q_{\exp}}$;
    if $temp > 0$ then
        $\hat{w} \leftarrow \log_\gamma \left(\frac{C}{D} \cdot temp\right)$;
        if $\hat{w} < w$ then
            InsertPositive($PQ, s, \hat{w}$);
        end if
    end if
end if
```

These calculations are done on Lines 17–31.

Now we consider how far to the left the interval $[w^i(s) - \delta, w^i(s))$ can extend, and thus determine $w^{i+1}(s)$. It is sufficient to consider the switching point $\hat{w}$ to prefer an action $a \neq a^i$ over a^i, provided that for all s' and all $w \in [\hat{w}, w^i(s))$, it holds that $w + r(s, a', s') \in [w^{j_{s'}+1}(s'), w^{j_{s'}}(s'))$. Similar to the proof of Theorem 4.19, the action a^i is preferred over another action $a \neq a^i$ as long as

$$Cw + Cv^i(s) + D\gamma^w v^i_{\exp}(s) \geq Cw + Cq^i(s, a) + D\gamma^w q^i_{\exp}(s, a)$$
$$Cv^i(s) + D\gamma^w v^i_{\exp}(s) \geq Cq^i(s, a) + D\gamma^w q^i_{\exp}(s, a).$$

where

$$q^i(s, a) = \sum_{s' \in S} P(s'|s, a)[r(s, a, s') + v^{j_{s'}}(s')]$$

and

$$q^i_{\exp}(s, a) = \sum_{s' \in S} P(s'|s, a)\gamma^{r(s,a,s')} v^{j_{s'}}_{\exp}(s').$$

Since a^i is optimal for w close to $w^i(s)$, it is impossible to have $v^i(s) < q^i(s, a)$ and $v^i_{\exp}(s) < q^i_{\exp}(s, a)$. If $v^i(s) \geq q^i(s, a)$ and $v^i_{\exp}(s) \geq q^i_{\exp}(s, a)$, then a^i is always preferred over a. Otherwise, the switching point to prefer a over a^i is

$$\hat{w} = \log_\gamma \left(\frac{C}{D} \cdot \frac{v^i(s) - q^i(s, a)}{q^i_{\exp}(s, a) - v^i_{\exp}(s)} \right).$$

Then the switching point to prefer an action other than a^i can be determined by considering all actions in A_s, and it is also the value for $w^{i+1}(s)$. The switching points are

Algorithm 4.9 Customized Insertion Operation for Algorithm 4.7

```
InsertPositive($PQ, s, w$)
if $w > 0$ then
    if IsMember($PQ, s$) then
        if GetKey($PQ, s$) $< w$ then
            IncreaseKey($PQ, s, w$);
        end if
    else
        Insert($PQ, s, w$);
    end if
end if
```

calculated on Lines 32–34. We actually determine all potential switching points where another action is preferred over the current one, but the closest one will be determined by the priority queue PQ (see below). Notice that we still need to make sure that $w + r(s, a, s') \in [w^{j_{s'}+1}(s'), w^{j_{s'}}(s'))$ for all $w \in [\hat{w}, w^i(s))$. We guarantee this by adding potential switching points for those w values that lead to known switching points, that is, w values such that $w + r(s, a, s')$ is a switching point for some states s, s' and action a. These switching points are taken into account on Lines 35–39.

Since we need to determine the optimal values and actions for $[w^{j_{s'}+1}(s'), w^{j_{s'}}(s'))$ before we determine those for $[w^{i+1}(s), w^i(s))$, we need to proceed in the order of decreasing w. Therefore, we use a priority queue PQ that stores pairs (s, w) where w is used as the key, and a larger value has a higher priority. The InsertPositive operation is a customized version of a regular insertion operation for priority queues and shown in Algorithm 4.9 (InsertPositive). The ExtractMax operation removes the entry with the largest key and returns it. The backward induction procedure terminates when the queue is empty.

4.5 Summary

In this chapter, we studied risk-sensitive planning with general utility functions. We took a state-augmentation approach, and showed that we can obtain aMD-optimal policies for finite horizon problems and approximate optimal values for infinite horizon problems with value iteration. We used deadline utility functions and a painted-blocks problem to demonstrate how this approach works. We showed the resulting functional value functions and greedy policies. We also obtained an exact algorithm for planning with one-switch utility functions, based on the results for linear, exponential, and general utility functions. We further showed the resulting value functions and the optimal policy for the painted-blocks problem, obtained using the exact algorithm.

CHAPTER V

PROBLEMS WITH ARBITRARY REWARDS

In this chapter, we consider risk-sensitive planning problems with arbitrary rewards. We study existence and finiteness conditions for the optimal values. Such conditions depend on properties of the utility function as well as the particular MDP model. These conditions are the prerequisites for the development of properties and computational procedures for problems with arbitrary rewards.

Recall that the value of a state $s \in S$ under a policy $\pi \in \Pi$ is defined as

$$v_U^\pi(s) = \lim_{T\to\infty} v_{U,T}^\pi(s) = \lim_{T\to\infty} E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right]. \tag{5.1}$$

The value exists if the finite horizon values converge on the extended real line, that is, to a finite value, positive infinity, or negative infinity. We therefore study the behavior of finite horizon values $v_{U,T}^\pi(s)$ as T approaches infinity.

In this chapter, we develop conditions under which the values exist for stationary policies. We also conjecture that under these sets of conditions, the values exist for all policies, and thus the optimal values exist for all policies. We then provide conditions under which the optimal values are finite, provided that the conjectures hold.

5.1 Results for the MER *Objective*

In general, MDPs can have both positive and negative (as well as zero) rewards. For the limit in Eq. (5.1) to exist, it is necessary that the finite horizon values $v_{U,T}^\pi(s)$ do not oscillate as T increases. This condition turns out to be related to the existence condition of values under the MER objective.

For an MDP with arbitrary rewards, the existence and finiteness conditions of values under the MER objective are associated with the positive and negative parts of the model. Following (Puterman, 1994), we define the positive part of a real number r to be $r^+ =$

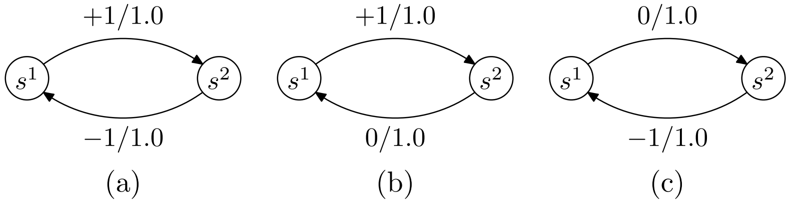


Figure 5.1: The example MDP from Figure 2.2 and its positive and negative parts

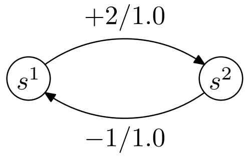


Figure 5.2: An example MDP with well-defined values but excluded by Condition 5.1

$\max(r, 0)$ and its negative part to be $r^- = \min(r, 0)$. We then obtain the positive part of an MDP by replacing every reward of the MDP with its positive part. We use $v^{+\pi}(s)$ to denote the values of the positive part of an MDP under policy $\pi \in \Pi$ under the MER objective, and $v^{+*}(s)$ to denote the optimal values of the positive part of the MDP under the MER objective. We define the negative part of an MDP and the values under the MER objective $v^{-\pi}(s)$ and $v^{-*}(s)$ in an analogous way. For later convenience, let

$$r^+_{\max} = \max_{P(s'|s,a)>0} r^+(s, a, s'), \quad r^-_{\min} = \min_{P(s'|s,a)>0} r^-(s, a, s'), \quad r_{\max} = \max\left(r^+_{\max}, -r^-_{\min}\right).$$

The following condition is sufficient for the MER values to exist for all policies (Puterman, 1994).

Condition 5.1 (One-Sided Finite Expected Rewards)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, at least one of $v^{+\pi}(s)$ and $v^{-\pi}(s)$ is finite.

Condition 5.1 is more general than Condition 2.5 (Negative Model) or Condition 2.7 (Positive Model), since, for example, Condition 2.5 implies that for all policies $\pi \in \Pi$ and all states $s \in S$, $v^{+\pi}(s) = 0$. In fact, Condition 5.1 is the weakest condition that we use in this chapter. Under this condition, for all policies $\pi \in \Pi$ and all states $s \in S$, the risk-neutral value $v^\pi(s)$ can be decomposed into the positive and negative parts as $v^\pi(s) = v^{+\pi}(s) + v^{-\pi}(s)$ (Puterman, 1994). Also under the same condition, for all states $s \in S$, the optimal value $v^*(s)$ exists, but it is not guaranteed to be finite (Puterman, 1994).

The MDPs in Figure 5.1(a) (also shown in Figure 2.2) and Figure 5.2 illustrate Condition 5.1. The MDP in Figure 5.1(a) does not satisfy Condition 5.1. The values of its states do not exist under its only policy π, as we have argued earlier. The positive and negative parts are shown in Figure 5.1(b) and Figure 5.1(c), respectively. Then, it is easy to see that $v^{+\pi}(s^1) = \infty$ and $v^{-\pi}(s^1) = -\infty$, which violates Condition 5.1 and illustrates that Condition 5.1 indeed rules out MDPs whose values do not exist under all policies.

The MDP in Figure 5.2 is another MDP that does not satisfy Condition 5.1. The values of its states, however, exist under its only policy π'. For example, an agent that starts in state s^1 receives the following sequence of rewards: $+2, -1, +2, -1, \ldots$, and consequently the following sequence of total rewards: $+2, +1, +3, +2, +4, +3, \ldots$, which converges toward positive infinity. Thus, the limit in Eq. (5.1) exists under π', and the value of state s^1 thus exists as well under π'. However, it is easy to see that $v^{+\pi'}(s^1) = \infty$ and $v^{-\pi'}(s^1) = -\infty$, which violates Condition 5.1 and demonstrates that Condition 5.1 is not a necessary condition for the values to exist under all policies.

According to our earlier discussion in Section 2.1, the optimal values need to be finite to do meaningful planning. For the optimal values to be finite, a set of conditions stronger than Condition 5.1 is needed. The following conditions are well-known in the literature (Feinberg, 2002).

Condition 5.2 (Finite Positive-Part Expected Rewards)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the value $v^{+\pi}(s)$ is finite.

Condition 5.3 (Finite Negative-Part Expected Rewards)**.** There exists a policy $\pi \in \Pi$ such that for all states $s \in S$, the value $v^{-\pi}(s)$ is finite.

If Condition 5.2 and Condition 5.3 hold, the optimal values are finite (Feinberg, 2002). If Condition 2.1 (Finite Model) holds, the optimal values are finite even if Π is replaced with Π^{SD} in Condition 5.2 and Condition 5.3, since there exists an SD-optimal policy under the MER objective (Puterman, 1994). In the rest of this chapter, we always assume that Condition 2.1 holds.

5.1.1 Structural Implications

Condition 5.1 (One-Sided Finite Expected Rewards) not only ensures the existence of values of states under all policies, but also has implications on the reward structure of the MDP. These structural implications make it interesting to consider Condition 5.1 when discussing risk-sensitive planning objectives in general.

Our main interest is on SR policies in this chapter. For a given SR policy π, the model is reduced to a Markov chain and the states can be classified into recurrent states and transient states. Let R^π denote the set of recurrent states under π. The set R^π can be partitioned into disjoint recurrent classes, denoted as R_i^π where $i \in I^\pi$ and I^π is the index set for recurrent classes under policy π. That is, if $i, j \in I^\pi$ but $i \neq j$, then $R_i^\pi \cap R_j^\pi = \varnothing$; and $\bigcup\limits_{i \in I^\pi} R_i^\pi = R^\pi$. Furthermore, for all states $s \in R_i^\pi$ and all states $s' \in R_j^\pi$ with $i \neq j$, $P^{s,\pi}(s_t = s') = 0$ for all epochs t.

For SR policies, we have the following results concerning recurrent states.

Lemma 5.1. *Assume that Condition 2.1 (Finite Model) and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an SR policy, and R_i^π be a recurrent class under π. For any state s that is recurrent under π, let $A_\pi(s) \subseteq A$ denote the set of actions whose probability is positive under the probability distribution $\pi(s, \cdot)$. Then exactly one of the following cases holds.*

a. *For all states $s \in R_i^\pi$, $v^\pi(s) = 0$, and for all actions $a \in A_\pi(s)$ and all states $s' \in S$ with $P(s'|s,a) > 0$, $r(s,a,s') = 0$.*

b. *For all states $s \in R_i^\pi$, $v^\pi(s) = \infty$, and for all actions $a \in A_\pi(s)$ and all states $s' \in S$ with $P(s'|s,a) > 0$, $r(s,a,s') \geq 0$. Moreover, there exists a transition (s,a,s') in the recurrent class R_i^π such that $P(s'|s,a) > 0$ and $r(s,a,s') > 0$.*

c. *For all states $s \in R_i^\pi$, $v^\pi(s) = -\infty$, and for all actions $a \in A_\pi(s)$ and all states $s' \in S$ with $P(s'|s,a) > 0$, $r(s,a,s') \leq 0$. Moreover, there exists a transition (s,a,s') in the recurrent class R_i^π such that $P(s'|s,a) > 0$ and $r(s,a,s') < 0$.*

Proof. Notice that once the agent enters the recurrent class R_i^π, it cannot reach any state outside of R_i^π. Therefore without loss of generality, we can assume that the MDP consists of a single recurrent class and no transient states.

We first show that under Condition 5.1, the rewards are either all nonnegative or all nonpositive for all valid transitions. Suppose otherwise: there exist a valid transition $(\bar{s}, \bar{a}, \bar{s}')$ with a positive reward $r(\bar{s}, \bar{a}, \bar{s}') > 0$ and a valid transition $(\tilde{s}, \tilde{a}, \tilde{s}')$ with a negative reward $r(\tilde{s}, \tilde{a}, \tilde{s}') < 0$. Then for the positive part of the model, the transition $(\bar{s}, \bar{a}, \bar{s}')$ can occur infinitely many times with a positive probability, thus receiving the positive reward $r(\bar{s}, \bar{a}, \bar{s}')$ infinitely many times. Therefore, $v^{+\pi}(s) = \infty$ for all states s. Similarly, we obtain that $v^{-\pi}(s) = -\infty$ for all state s, which contradicts Condition 5.1.

Therefore, part (a) follows when only zero rewards are possible. Part (b) follows when there exists a positive reward, and part (c) follows when there exists a negative reward. □

According to Lemma 5.1, the index set I^π can be further divided into three disjoint classes $I_0^\pi, I_+^\pi, I_-^\pi$, such that for all recurrent classes R_i^π under π,

- if $i \in I_0^\pi$, then for all states $s \in R_i^\pi$, $v^\pi(s) = 0$;
- if $i \in I_+^\pi$, then for all states $s \in R_i^\pi$, $v^\pi(s) = \infty$; and
- if $i \in I_-^\pi$, then for all states $s \in R_i^\pi$, $v^\pi(s) = -\infty$.

We refer to these recurrent classes as zero, positive, and negative, respectively. Moreover, we use a superscript s to indicate the subset of indices of recurrent classes that can be reached from the state s, that is, we use notations $I^{s,\pi}, I_0^{s,\pi}, I_+^{s,\pi}, I_-^{s,\pi}$. We also use $R^{s,\pi} \subseteq R^\pi$ to denote the set of recurrent states that are reachable from state s under π.

The transient states can be similarly classified into three types according to the recurrent classes they can enter. We denote the set of transient states that can be reached from a state s under policy π as $T^{s,\pi}$. The following lemma states this classification for all states.

Lemma 5.2. *Assume that Condition 2.1 (Finite Model) and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an SR policy, and s be a state.*

a. *If $v^\pi(s)$ is finite, then for any s' that is reachable from s, $v^\pi(s')$ is finite. In particular, if s' is recurrent, then $v^\pi(s') = 0$.*

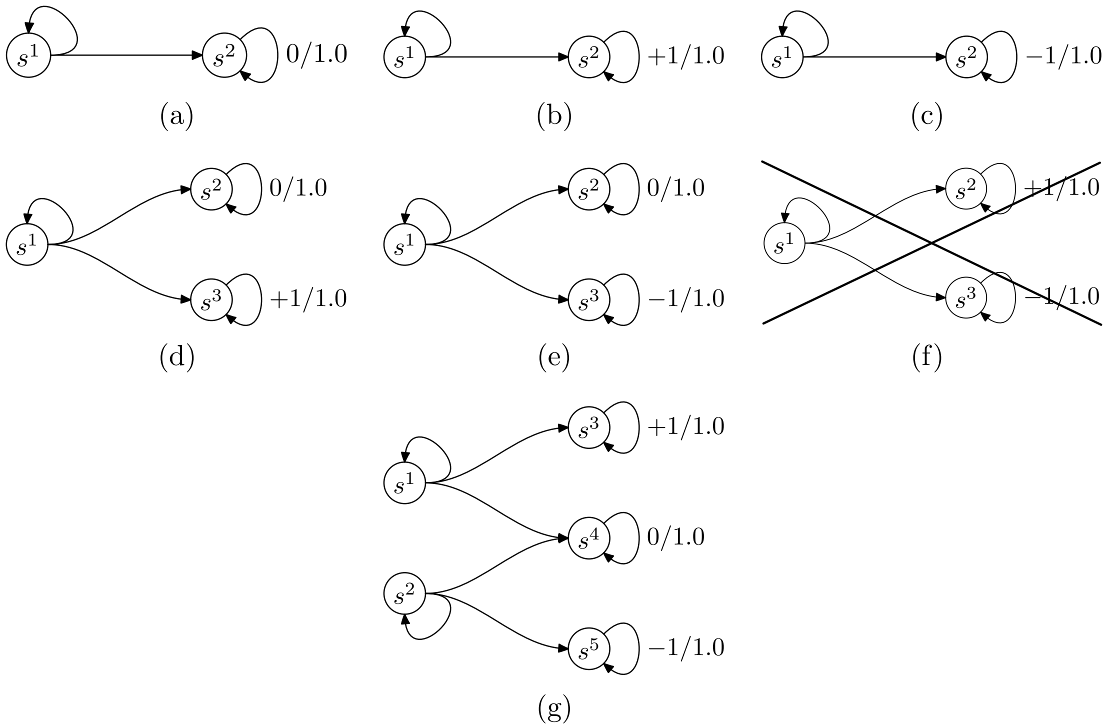


Figure 5.3: Example MDPs that illustrate Lemma 5.2

b. *If $v^{\pi}(s) = \infty$, then for any s' that is reachable from s under π, $v^{\pi}(s') = \infty$ or $v^{\pi}(s')$ is finite. In particular, if s' is recurrent, then $v^{\pi}(s') = \infty$ or $v^{\pi}(s') = 0$.*

c. *If $v^{\pi}(s) = -\infty$, then for any s' that is reachable from s under π, $v^{\pi}(s') = -\infty$ or $v^{\pi}(s')$ is finite. In particular, if s' is recurrent, then $v^{\pi}(s') = -\infty$ or $v^{\pi}(s') = 0$.*

Proof. We prove this result by contradiction. Notice that the values exist for all policies and all states under Condition 5.1.

For part (a), suppose otherwise: $v^{\pi}(s)$ is finite, and there exists $s'' \in T^{s,\pi} \cup R^{s,\pi}$ such that $v^{\pi}(s'') = \infty$ or $v^{\pi}(s'') = -\infty$. If $v^{\pi}(s'') = \infty$, then it must be that $v^{+\pi}(s'') = \infty$ and $v^{-\pi}(s'')$ is finite. But in this case, it follows that $v^{+\pi}(s) = \infty$ since there is a positive probability p to reach s'' from s under π, and thus $v_T^{+\pi}(s) \geq p v_T^{+\pi}(s'')$ for all $T \in \mathbb{N}$. Then it follows from Condition 5.1 that $v^{-\pi}(s)$ is finite and $v^{\pi}(s) = v^{+\pi}(s) + v^{-\pi}(s) = +\infty$, which contradicts our assumption that $v^{\pi}(s)$ is finite. Therefore $v^{\pi}(s'') \neq \infty$. Similarly, we have $v^{\pi}(s'') \neq -\infty$. Therefore, the result holds if $v^{\pi}(s)$ is finite.

Parts (b) and (c) are similar. □

Lemma 5.2 is illustrated in Figure 5.3. In the simplest case, the agent can only reach one recurrent class under an SR policy, and only zero, positive, or negative rewards are received in recurrent states, as illustrated by Figure 5.3(a), Figure 5.3(b), and Figure 5.3(c), respectively. It is possible to reach more than one recurrent classes starting from the same state under an SR policy (Figure 5.3(d) or Figure 5.3(e)), but these recurrent classes cannot contain both positive and negative ones at the same time, therefore the case shown in Figure 5.3(f) is excluded by Condition 5.1. There can be both positive and negative recurrent classes at the same time, but they can only be reached starting from different states, as shown in Figure 5.3(g).

5.2 A Template for the Proofs of Existence Results

To show that the limit of $v^{\pi}_{U,T}(s)$ as T approaches infinity exists, we consider the accumulated rewards before and after the agent enters a recurrent state separately.

Let τ be the random variable indicating the epoch of entering R^{π}, that is, $s_{\tau} \in R^{\pi}$ and $s_{\tau-1} \notin R^{\pi}$ if $s_0 = s \notin R^{\pi}$, and $\tau = 0$ otherwise. It is necessary to distinguish the horizon when considering finite horizon problems. The reason is as follows. Consider horizons T_1 and T_2 where $T_1 \neq T_2$. The random variable τ for the T_1-horizon problem can take values from 0 to T_1, while the random variable τ for the T_2-horizon problem can take values from 0 to T_2. To avoid confusion, we use $\tau(T_1)$ and $\tau(T_2)$ to denote the two random variables, respectively. By τ alone, we mean the random variable in an infinite horizon problem. Also let w_{τ} and $w_{\tau(T)}$ be the total rewards up to τ and $\tau(T)$, respectively.

The finite horizon values can be decomposed as follows:

$$\begin{aligned}
v^{\pi}_{U,T}(s) = E^{s,\pi}\left[U\left(\sum_{t=0}^{T-1} r_t\right)\right] &= E^{s,\pi}\left[U(w_T)\right] \\
&= E^{s,\pi}[U(w_T)|s_T \notin R^{\pi}] \cdot P^{s,\pi}(s_T \notin R^{\pi}) + E^{s,\pi}[U(w_T)|s_T \in R^{\pi}] \cdot P^{s,\pi}(s_T \in R^{\pi}) \\
&= E^{s,\pi}[U(w_T)|s_T \notin R^{\pi}] \cdot P^{s,\pi}(s_T \notin R^{\pi}) \\
&\quad + E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^{\pi}] \cdot P^{s,\pi}(s_T \in R^{\pi}) \\
&\quad + E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^{\pi}] \cdot P^{s,\pi}(s_T \in R^{\pi}). \qquad (5.2)
\end{aligned}$$

The first term is the contribution of those trajectories not entering a recurrent state, the second term is the contribution of those parts of the trajectories up to the point when a recurrent state is entered, and the third term is the contribution of those parts of the trajectories after a recurrent state is entered. Therefore, the value $v_U^\pi(s)$ exists if the limit of each term as $T \rightarrow \infty$ exists.

We first consider the third term since it is the only term related to the behavior after a recurrent state is entered. This term can be decomposed further according to Lemma 5.2.

$$\begin{aligned}
&E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&= \sum_{i \in I^\pi} E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi) \\
&= \sum_{i \in I_0^\pi} E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi) \\
&\quad + \sum_{i \in I_+^\pi} E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi) \\
&\quad + \sum_{i \in I_-^\pi} E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi),
\end{aligned}$$

where at most one of I_+^π and I_-^π can be non-empty.

If $i \in I_0^\pi$, only zero rewards are possible after a recurrent state is entered. Therefore, $w_T = w_{\tau(T)}$ and $E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi) = 0$.

If $i \in I_+^\pi$, only nonnegative rewards are possible after a recurrent state is entered. Therefore, $w_T \geq w_{\tau(T)}$. We have

$$\begin{aligned}
&E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi) \\
&= \sum_{t=1}^{T} E^{s,\pi}[U(w_T) - U(w_t)|s_T \in R_i^\pi, s_{t-1} \notin R^\pi, s_t \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi, s_{t-1} \notin R^\pi, s_t \in R_i^\pi) \\
&= \sum_{t=1}^{T} E^{s,\pi}[U(w_T) - U(w_t)|s_{t-1} \notin R^\pi, s_t \in R_i^\pi] \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R_i^\pi).
\end{aligned}$$

For a fixed t, the value $E^{s,\pi}[U(w_T) - U(w_t)|s_{t-1} \notin R^\pi, s_t \in R_i^\pi] \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R_i^\pi)$ is monotonically nondecreasing with T. Consequently, the value $E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R_i^\pi] \cdot P^{s,\pi}(s_T \in R_i^\pi)$ is also monotonically nondecreasing with T since each term is nonnegative and monotonically nondecreasing with T. Again, its limit as $T \rightarrow \infty$ exists (the limit may be positive infinity). If the utility function is bounded from above, the limit is finite; otherwise, the limit is positive infinity.

If $i \in I^\pi_-$, the limit also exists for a reason similar to the case of $i \in I^\pi_+$ (but the limit can be negative infinity, instead). If the utility function is bounded from below, the limit is finite; otherwise, the limit is negative infinity.

Therefore, the limit of the summation of these terms over $i \in I^\pi$ also exists. We summarize the above discussion in the following lemma.

Lemma 5.3. *Assume that Condition 2.1 (Finite Model) and Condition 5.1 (One-Sided Finite Expected Rewards) hold. For all policies $\pi \in \Pi$ and all states $s \in S$, the following limit*

$$\lim_{T\to\infty} E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

exists. Moreover, if $v^\pi(s)$ is finite, the limit is zero; if $v^\pi(s) = \infty$, the limit is finite if the utility function is bounded from above, and positive infinity otherwise; and if $v^\pi(s) = -\infty$, the limit is finite if the utility function is bounded from below, and negative infinity otherwise.

□

In the rest of this chapter, we consider the limits of the first two terms of Eq. (5.2) under different sets of conditions. Under these conditions, we also identify whether the third term is finite or infinite, therefore fully specify the existence and finiteness properties of the value $v^\pi_U(s)$. In fact, we try to establish that under these different conditions, if the first term converges to zero as $T \to \infty$ then the second term converges to a finite value, and if the first term converges to a finite number different from zero or infinity then the second term converges to infinity.

For the risk-neutral utility function (and equivalently, identity or linear utility functions), the following result holds.

Lemma 5.4. *Assume that Condition 2.1 (Finite Model) and Condition 5.1 (One-Sided Finite Expected Rewards) hold. It holds that*

$$\lim_{T\to\infty} E^{s,\pi}[w_T|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0$$

and the limit

$$\lim_{T\to\infty} E^{s,\pi}[w_{\tau(T)}|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

is finite.

Proof. First notice that it is well-known (Kemeny and Snell, 1960) that there exists $a \in (0, +\infty)$ and $\rho \in (0, 1)$ such that

$$P^{s,\pi}(s_T \notin R^\pi) \le a\rho^T.$$

Since $|w_T| \le r_{\max}T$, we have

$$\begin{aligned}\left|E^{s,\pi}[w_T|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi)\right| &= \left|E^{s,\pi}[w_T|s_T \notin R^\pi]\right| \cdot P^{s,\pi}(s_T \notin R^\pi) \\ &\le r_{\max}T \cdot P^{s,\pi}(s_T \notin R^\pi) \le r_{\max}T \cdot a\rho^T = ar_{\max}T\rho^T.\end{aligned}$$

Since $\lim\limits_{T\to\infty} T\rho^T = 0$, it follows that

$$\lim_{T\to\infty} E^{s,\pi}[w_T|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0.$$

For the second part of the lemma, we have the following decomposition

$$\begin{aligned}&E^{s,\pi}[w_{\tau(T)}|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\ &= \sum_{t=1}^{T} E^{s,\pi}[w_t|s_T \in R^\pi, s_{t-1} \notin R^\pi, s_t \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi, s_{t-1} \notin R^\pi, s_t \in R^\pi) \\ &= \sum_{t=1}^{T} E^{s,\pi}[w_t|s_{t-1} \notin R^\pi, s_t \in R^\pi] \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R^\pi).\end{aligned}$$

Therefore,

$$\begin{aligned}&\lim_{T\to\infty} E^{s,\pi}[w_{\tau(T)}|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\ &\qquad = \sum_{t=1}^{\infty} E^{s,\pi}[w_t|s_{t-1} \notin R^\pi, s_t \in R^\pi] \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R^\pi).\end{aligned} \tag{5.3}$$

For each term of the above summation, we have

$$\begin{aligned}&\left|E^{s,\pi}[w_t|s_{t-1} \notin R^\pi, s_t \in R^\pi] \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R^\pi)\right| \\ &= \left|E^{s,\pi}[w_t|s_{t-1} \notin R^\pi, s_t \in R^\pi]\right| \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R^\pi) \\ &\le r_{\max}t \cdot P^{s,\pi}(s_{t-1} \notin R^\pi, s_t \in R^\pi) \le r_{\max}t \cdot P^{s,\pi}(s_{t-1} \notin R^\pi) \le r_{\max}t \cdot a\rho^{t-1} = \frac{ar_{\max}}{\rho} \cdot t\rho^t.\end{aligned}$$

Since $\sum_{t=1}^{\infty} t\rho^t = \frac{\rho}{(1-\rho)^2}$, the limit in Eq. (5.3) exists and is finite. □

In the next section, we discuss exponential utility functions. Linear and exponential utility functions are "landmarks" in the sense that the existence and finiteness properties of MDPs with general risk-sensitive utility functions are characterized by the existence and finiteness properties of these MDPs with linear or exponential utility functions that dominate these utility functions in the sense of the big O notation.

5.3 Exponential Utility Functions

We discuss convex and concave exponential utility functions separately for MDPs with both positive and negative rewards. Similar to the risk-neutral case, we also need to refer to the values of the positive and negative parts. We use $v_{\text{exp}}^{+\pi}(s)$ to denote the values of the positive part of an MDP with exponential utility functions under policy $\pi \in \Pi$ and $v_{\text{exp}}^{+*}(s)$ to denote the optimal values of the positive part. We define the values $v_{\text{exp}}^{-\pi}(s)$ and $v_{\text{exp}}^{-*}(s)$ in an analogous way.

In general, Condition 5.1 (One-Sided Finite Expected Rewards) is not sufficient to ensure the existence of values for a given policy under the $\mathsf{MEU}_{\text{exp}}$ objective. We need to have stronger conditions for the existence of values. To show an example illustrating the necessity of stronger conditions, we first need the following lemma for finite horizon values.

Lemma 5.5. *Assume that Condition 2.1 (Finite Model) holds. For a given $\pi \in \Pi^{SR}$, let D_π be a matrix whose (s,s')-entry is*

$$D_\pi(s,s') = \sum_{a\in A_s} \pi(s,a)P(s'|s,a)\gamma^{r(s,a,s')}.$$

Let D_π^T indicate the T-th power of D_π. Then the (s,s')-entry of D_π^T is

$$D_\pi^T(s,s') = E^{s,\pi}[\gamma^{w_T}|s_T = s'] \cdot P^{s,\pi}(s_T = s').$$

The finite horizon values can then be calculated as

$$v_{\text{exp},T}^{\pi}(s) = \iota \cdot \sum_{s'\in S} D_\pi^T(s,s').$$

Proof. We prove the result by induction. Suppose $T = 1$. For all $s' \in S$, we have

$$\begin{aligned}&E^{s,\pi}\left[\gamma^{w_T}\middle|\, s_T = s'\right] \cdot P^{s,\pi}(s_T = s')\\ &= E_a^{s,\pi}\left[E^{s,\pi}\left[\gamma^{w_T}\middle|\, s_T = s', a_{T-1} = a\right]\middle| s_T = s'\right] \cdot P^{s,\pi}(s_T = s')\end{aligned}$$

$$= E_a^{s,\pi}\left[\gamma^{r(s,a,s')}\middle|\, s_T = s'\right] \cdot P^{s,\pi}(s_T = s')$$
$$= \sum_{a \in A_s} \pi(s,a) P(s'|s,a) \gamma^{r(s,a,s')}.$$

Therefore, the result holds for $T = 1$.

Suppose $T \geq 1$ and the result holds. We now show that the result then also holds for $T+1$. We have

$$\begin{aligned}
&E^{s,\pi}\left[\gamma^{w_{T+1}} \middle|\, s_{T+1} = s'\right] \cdot P^{s,\pi}(s_{T+1} = s') \\
&= E_{s'',a}^{s,\pi}\left[E^{s,\pi}\left[\gamma^{w_{T+1}} \middle|\, s_{T+1} = s', s_T = s'', a_T = a\right] \middle|\, s_{T+1} = s'\right] \cdot P^{s,\pi}(s_{T+1} = s') \\
&= E_{s'',a}^{s,\pi}\left[E^{s,\pi}\left[\gamma^{w_T} \cdot \gamma^{r(s'',a,s')} \middle|\, s_{T+1} = s', s_T = s'', a_T = a\right] \middle|\, s_{T+1} = s'\right] \cdot P^{s,\pi}(s_{T+1} = s') \\
&= E_{s'',a}^{s,\pi}\left[E^{s,\pi}\left[\gamma^{w_T} \middle|\, s_T = s''\right] \cdot \gamma^{r(s'',a,s')} \middle|\, s_{T+1} = s'\right] \cdot P^{s,\pi}(s_{T+1} = s') \\
&= \sum_{s'' \in S} \sum_{a \in A_{s''}} P^{s,\pi}(s_T = s'') \cdot \pi(s'',a) P(s'|s'',a) \cdot E^{s,\pi}\left[\gamma^{w_T} \middle|\, s_T = s''\right] \cdot \gamma^{r(s'',a,s')} \\
&= \sum_{s'' \in S} P^{s,\pi}(s_T = s'') \cdot E^{s,\pi}\left[\gamma^{w_T} \middle|\, s_T = s''\right] \cdot \sum_{a \in A_{s''}} \pi(s'',a) P(s'|s'',a) \gamma^{r(s'',a,s')} \\
&= \sum_{s'' \in S} D_\pi^T(s,s'') \cdot D_\pi(s'',s') = D_\pi^{T+1}(s,s').
\end{aligned}$$

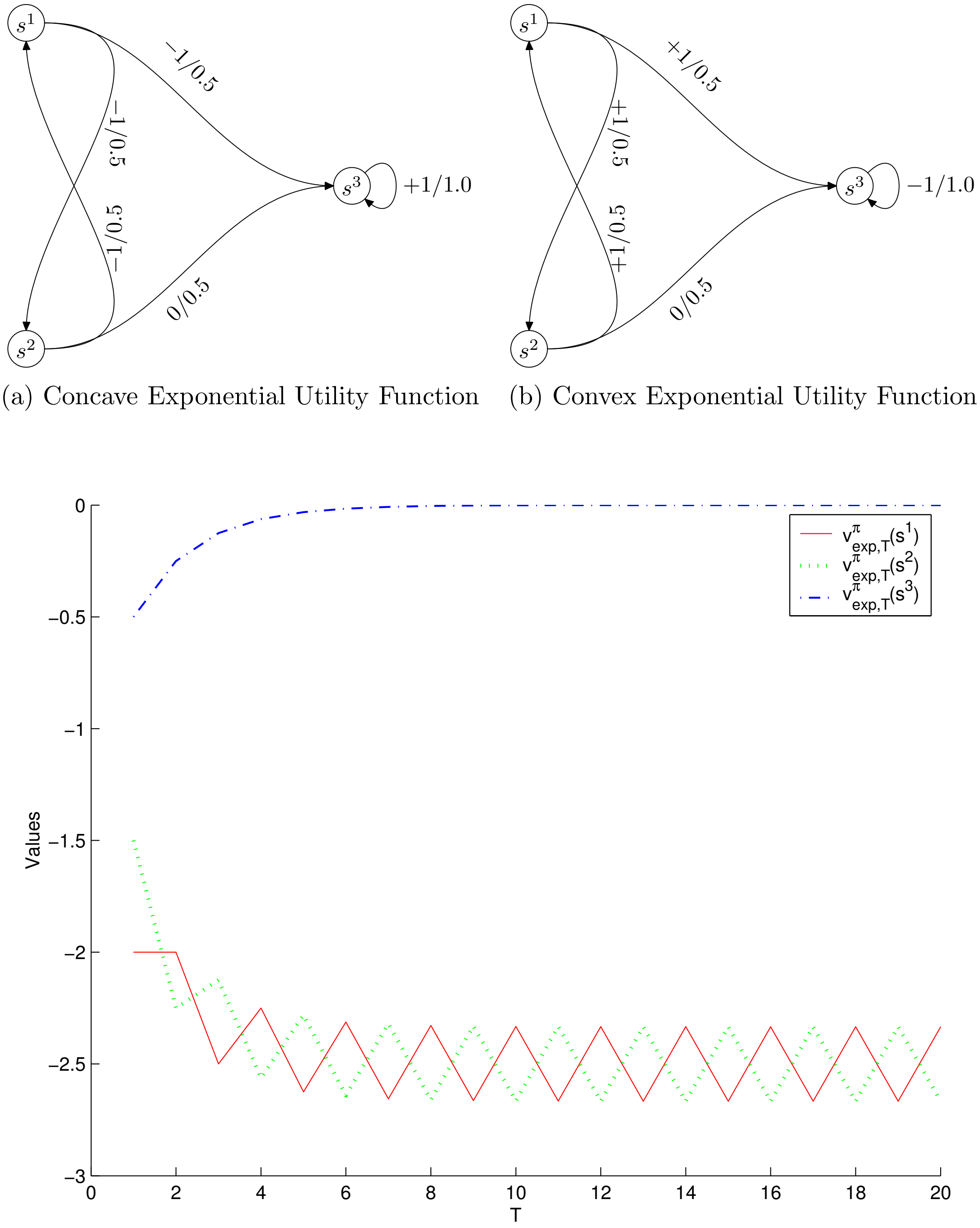
Therefore, the result holds. □

Now we present an example that satisfies Condition 5.1 (One-Sided Finite Expected Rewards), but the values do not exist. Suppose the utility function is $U(w) = -\left(\frac{1}{2}\right)^w$. The example MDP is shown in Figure 5.4(a), which has only one policy π. Condition 5.1 holds since only positive rewards are received in s^3, which is the only recurrent state in this MDP. According to Lemma 5.5, the values can be obtained by calculating the powers of

$$D_\pi = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} \end{pmatrix}.$$

We can verify that

$$D_\pi^T = \begin{pmatrix} \frac{1}{2}(1+(-1)^T) & \frac{1}{2}(1-(-1)^T) & \frac{3}{2} - \frac{1}{6}(-1)^T - \frac{4}{3}\left(\frac{1}{2}\right)^T \\ \frac{1}{2}(1-(-1)^T) & \frac{1}{2}(1+(-1)^T) & \frac{3}{2} + \frac{1}{6}(-1)^T - \frac{5}{3}\left(\frac{1}{2}\right)^T \\ 0 & 0 & \left(\frac{1}{2}\right)^T \end{pmatrix}.$$



(a) Concave Exponential Utility Function

(b) Convex Exponential Utility Function

(c) The First 20 Finite-Horizon Values

Figure 5.4: MDPs satisfying Condition 5.1 but without expected utilities

The finite horizon values then are

$$v^{\pi}_{\exp,T}(s^1) = -\frac{5}{2} + \frac{1}{6}(-1)^T + \frac{4}{3}\left(\frac{1}{2}\right)^T,$$
$$v^{\pi}_{\exp,T}(s^2) = -\frac{5}{2} - \frac{1}{6}(-1)^T + \frac{5}{3}\left(\frac{1}{2}\right)^T,$$
$$v^{\pi}_{\exp,T}(s^3) = -\left(\frac{1}{2}\right)^T.$$

Therefore, the limit does not exist for s^1 or s^2. We plot the first 20 finite horizon values in Figure 5.4(c). With the same D_π matrix, we also have an example for the convex utility function $U(w) = 2^w$, as shown in Figure 5.4(b).

The conditions that can eliminate the above case are different for concave and convex exponential utility functions. Before considering them separately, we now present some results that hold for both concave and convex exponential utility functions. These results are the basis of our proofs for the existence of values under the $\mathsf{MEU}_{\exp}$ objective.

As we discussed earlier in Section 5.2, we need to consider the contributions of the parts of trajectories up to the point when a recurrent state is entered. First, we can calculate such contributions for any given finite horizon according to the following lemma, in a way similar to Lemma 5.5.

Lemma 5.6. *Assume that Condition 2.1 (Finite Model) holds. For a given $\pi \in \Pi^{SR}$, let $\hat{D}_\pi$ be a matrix whose (s, s')-entry is*

$$\hat{D}_\pi(s,s') = \begin{cases} \sum\limits_{a\in A_s} \pi(s,a)P(s'|s,a)\gamma^{r(s,a,s')}, & s \notin R^\pi; \\ 1, & s = s', s \in R^\pi; \\ 0, & s \neq s', s \in R^\pi. \end{cases}$$

Let $\hat{D}^T_\pi$ indicate the T-th power of $\hat{D}_\pi$. Then the (s, s')-entry of $\hat{D}^T_\pi$ is

$$\hat{D}^T_\pi(s,s') = \begin{cases} E^{s,\pi}\left[\gamma^{w_T}|s_T = s'\right] \cdot P^{s,\pi}(s_T = s'), & s, s' \notin R^\pi; \\ E^{s,\pi}\left[\gamma^{w_{\tau(T)}}|s_{\tau(T)} = s'\right] \cdot P^{s,\pi}(s_{\tau(T)} = s'), & s \notin R^\pi, s' \in R^\pi; \\ 1, & s = s', s \in R^\pi; \\ 0, & s \neq s', s \in R^\pi. \end{cases}$$

Proof. Notice that we can order the states such that $\hat{D}_\pi$ can be written as

$$\hat{D}_\pi = \begin{pmatrix} A_\pi & B_\pi \\ \mathbf{0} & \mathbf{1} \end{pmatrix}$$

where $\mathbf{1}$ is an identity matrix and $\mathbf{0}$ is a zero matrix. The top half submatrix corresponds to transient states $s \in T^\pi$ and the bottom half submatrix corresponds to recurrent states $s \in R^\pi$. Thus we have

$$\hat{D}_\pi^T = \begin{pmatrix} A_\pi^T & A_\pi^{T-1} B_\pi + \cdots + B_\pi \\ \mathbf{0} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} A_\pi^T & \left(\sum_{t=0}^{T-1} A_\pi^t \right) \cdot B_\pi \\ \mathbf{0} & \mathbf{1} \end{pmatrix}$$

Therefore, if $s \in R^\pi$, the result holds.

Moreover, if $s \notin R^\pi$ and $s' \notin R^\pi$, Lemma 5.5 shows that the result holds.

Next, we prove by induction that if $s \notin R^\pi$ and $s' \in R^\pi$, the result also holds. Suppose $T = 1$. It then must be the case that $\tau(T) = 1$. Therefore, the result also follows from Lemma 5.5.

Suppose that the result holds for $T \geq 1$. We now show that the result also holds for $T + 1$. Notice that when the current decision epoch is T, we have

$$\begin{aligned} \hat{D}_\pi^T(s, s') &= E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \middle| s_{\tau(T)} = s' \right] \cdot P^{s,\pi}(s_{\tau(T)} = s') \\ &= \sum_{t=1}^{T} E^{s,\pi}\left[\gamma^{w_t} \middle| s_t = s', s_{t-1} \notin R^\pi \right] \cdot P^{s,\pi}(s_t = s', s_{t-1} \notin R^\pi). \end{aligned} \tag{5.4}$$

Therefore, when the current decision epoch is $T + 1$, we have

$$\begin{aligned} & E^{s,\pi}\left[\gamma^{w_{\tau(T+1)}} \middle| s_{\tau(T+1)} = s' \right] \cdot P^{s,\pi}(s_{\tau(T+1)} = s') \\ &\quad = \sum_{t=1}^{T+1} E^{s,\pi}\left[\gamma^{w_t} \middle| s_t = s', s_{t-1} \notin R^\pi \right] \cdot P^{s,\pi}(s_t = s', s_{t-1} \notin R^\pi) \\ &\quad = \hat{D}_\pi^T(s, s') + E^{s,\pi}\left[\gamma^{T+1} \middle| s_{T+1} = s', s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi). \end{aligned} \tag{5.5}$$

Hence we have

$$\begin{aligned} & E^{s,\pi}\left[\gamma^{T+1} \middle| s_{T+1} = s', s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi) \\ &= E^{s,\pi}_{s'',a}\left[E^{s,\pi}\left[\gamma^{T+1} \middle| s_{T+1} = s', s_T \notin R^\pi, s_T = s'', a_T = a \right] \middle| s_{T+1} = s', s_T \notin R^\pi \right] \\ &\qquad \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi) \\ &= E^{s,\pi}_{s'',a}\left[E^{s,\pi}\left[\gamma^T \cdot \gamma^{r(s'',a,s')} \middle| s_{T+1} = s', s_T \notin R^\pi, s_T = s'', a_T = a \right] \middle| s_{T+1} = s', s_T \notin R^\pi \right] \\ &\qquad \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi) \\ &= E^{s,\pi}_{s'',a}\left[E^{s,\pi}\left[\gamma^T \middle| s_T = s'' \right] \cdot \gamma^{r(s'',a,s')} \middle| s_{T+1} = s', s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi) \end{aligned}$$

$$= \sum_{s'' \notin R^\pi} \sum_{a \in A_{s''}} P^{s,\pi}(s_T = s'') \cdot \pi(s'', a) P^{s,\pi}(s'|s'', a) \cdot E^{s,\pi}\left[\gamma^{w_T} \middle| s_T = s''\right] \cdot \gamma^{r(s'',a,s')}$$

$$= \sum_{s'' \notin R^\pi} P^{s,\pi}(s_T = s'') \cdot E^{s,\pi}\left[\gamma^{w_T} \middle| s_T = s''\right] \cdot \sum_{a \in A_{s''}} \pi(s'', a) P^{s,\pi}(s'|s'', a) \gamma^{r(s'',a,s')}$$

$$= \sum_{s'' \notin R^\pi} \hat{D}_\pi^T(s, s'') \cdot \hat{D}_\pi(s'', s').$$

Since $\hat{D}_\pi(s'', s') = 1$ if $s'' = s'$ and $\hat{D}_\pi(s'', s') = 0$ if $s'' \neq s'$, it holds that

$$\begin{aligned}
&E^{s,\pi}\left[\gamma^{w_{\tau(T+1)}} \middle| s_{\tau(T+1)} = s'\right] \cdot P^{s,\pi}(s_{\tau(T+1)} = s') \\
&\quad = \hat{D}_\pi^T(s, s') + E^{s,\pi}\left[\gamma^{T+1} \middle| s_{T+1} = s', s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_{T+1} = s', s_T \notin R^\pi) \\
&\quad = \hat{D}_\pi^T(s, s') + \sum_{s'' \notin R^\pi} \hat{D}_\pi^T(s, s'') \cdot \hat{D}_\pi(s'', s') \\
&\quad = \sum_{s'' \in S} \hat{D}_\pi^T(s, s'') \cdot \hat{D}_\pi(s'', s') = \hat{D}_\pi^{T+1}(s, s'),
\end{aligned}$$

Therefore, the result holds for all T. □

Next, we have the following results concerning the first two terms of Eq. (5.2). These results are analogous to Lemma 5.4, the result for linear utility functions.

Define $\hat{D}_{s,\pi}$, the restriction of $\hat{D}_\pi$ to s, as

$$\hat{D}_{s,\pi}(s', s'') = \begin{cases} \hat{D}_\pi(s', s''), & s' \in R^{s,\pi} \cup T^{s,\pi}; \\ 0, & \text{otherwise.} \end{cases}$$

Lemma 5.6 also holds for all $s', s'' \in R^{s,\pi} \cup T^{s,\pi}$ since other states are irrelevant.

Lemma 5.7. *Assume that Condition 2.1 (Finite Model) holds. Let $\pi \in \Pi^{SR}$. For all $s \notin R^\pi$,*

$$\lim_{T \to \infty} E^{s,\pi}\left[\gamma^{w_T} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0$$

if and only if the limit

$$\lim_{T \to \infty} E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi)$$

exists and is finite.

Proof. Notice that for all $s \notin R^\pi$,

$$E^{s,\pi}\left[\gamma^{w_T} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = \sum_{s' \in T^{s,\pi}} E^{s,\pi}\left[\gamma^{w_T} \middle| s_T = s'\right] \cdot P^{s,\pi}(s_T = s')$$

$$= \sum_{s' \in T^{s,\pi}} \hat{D}^T_{s,\pi}(s, s')$$

and

$$\begin{aligned} E^{s,\pi}\left[\gamma^{w_\tau(T)} \,|\, s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi) &= \sum_{s' \in R^{s,\pi}} E^{s,\pi}\left[\gamma^{w_\tau(T)} \,|\, s_T = s'\right] \cdot P^{s,\pi}(s_T = s') \\ &= \sum_{s' \in R^{s,\pi}} \hat{D}^T_{s,\pi}(s, s'). \end{aligned}$$

We therefore consider the matrix $\hat{D}_{s,\pi}$. We can order the states such that

$$\hat{D}_{s,\pi} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & A_{s,\pi} & B_{s,\pi} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}$$

where $\mathbf{1}$ is an identity matrix, and $\mathbf{0}$ is a zero matrix, and

$$\hat{D}^T_{s,\pi} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & A^T_{s,\pi} & A^{T-1}_{s,\pi} \cdot B_{s,\pi} + \cdots + B_{s,\pi} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & A^T_{s,\pi} & \left(\sum_{t=0}^{T-1} A^t_{s,\pi}\right) \cdot B_{s,\pi} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}. \tag{5.6}$$

For the matrix $A_{s,\pi}$, there exists a matrix $E_{s,\pi}$ such that

$$A_{s,\pi} = E_{s,\pi} J_{s,\pi} E^{-1}_{s,\pi},$$

where $J_{s,\pi}$ is the Jordan canonical form with Jordan blocks J_i of sizes n_i for $i = 1, 2, \ldots, m$

$$J_{s,\pi} = \begin{pmatrix} J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_m \end{pmatrix}$$

and for each $i = 1, 2, \ldots, m$, $J_i = \begin{pmatrix} \lambda_i \end{pmatrix}$ if $n_i = 1$, and

$$J_i = \begin{pmatrix} \lambda_i & 1 & & \\ & \lambda_i & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{pmatrix}$$

if $n_i > 1$. We ignore zero elements in these matrices by convention. Moreover, it holds that

$$A^T_{s,\pi} = E_{s,\pi} J^T_{s,\pi} E^{-1}_{s,\pi} = E_{s,\pi} \begin{pmatrix} J_1^T & & & \\ & J_2^T & & \\ & & \ddots & \\ & & & J_m^T \end{pmatrix} E^{-1}_{s,\pi},$$

where for each $i = 1, 2, \ldots, m$, $J_i^T = \left(\lambda_i^T\right)$ if $n_i = 1$, and

$$J_i^T = \begin{pmatrix} \lambda_i^T & T\lambda_i^{T-1} & \cdots & \frac{T!}{(T-n_i+1)!}\lambda_i^{T-n_i+1} \\ & \lambda_i^T & \ddots & \frac{T!}{(T-n_i+2)!}\lambda_i^{T-n_i+2} \\ & & \ddots & T\lambda_i^{T-1} \\ & & & \lambda_i^T \end{pmatrix}$$

if $n_i > 1$. Therefore, the (s, s') element of $A^T_{s,\pi}$, where $s' \in T^{s,\pi}$, is a linear combination of entries in J_i^T as follows

$$\hat{D}^T_{s,\pi}(s, s') = \sum_{i=1}^{m} \sum_{j=0}^{n_i-1} a_{ij}^{s'} \cdot \frac{T!}{(T-j)!} \cdot \lambda_i^{T-j}, \tag{5.7}$$

where we use the convention that $\frac{1}{(-n)!} = 0$ for all $n = 1, 2, \ldots$.

Now we consider the first part of the result. Since

$$E^{s,\pi}\left[\gamma^{w_T} \middle| \, s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = \sum_{s' \in T^{s,\pi}} \hat{D}^T_{s,\pi}(s, s')$$

and each term is positive,

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma^{w_T} \middle| \, s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0$$

if and only if for all $s' \in T^{s,\pi}$,

$$\lim_{T\to\infty} \hat{D}^T_{s,\pi}(s, s') = 0. \tag{5.8}$$

Then according to Eq. (5.7), the above limit converges to zero as $T \to \infty$ if and only if for all i, j

$$|\lambda_i| < 1 \qquad \text{or} \qquad a_{ij}^{s'} = 0. \tag{5.9}$$

In this case, for $s'' \in R^{s,\pi}$, according to Eq. (5.6), it holds that

$$\begin{aligned} \hat{D}^T_{s,\pi}(s, s'') &= \sum_{s' \in T^{s,\pi}} \left(\sum_{t=0}^{T-1} \hat{D}^t_{s,\pi}(s, s') \right) \cdot \hat{D}_{s,\pi}(s', s'') \\ &= \sum_{s' \in T^{s,\pi}} \left(\sum_{t=0}^{T-1} \sum_{i=1}^{m} \sum_{j=0}^{n_i-1} a_{ij}^{s'} \cdot \frac{t!}{(t-j)!} \cdot \lambda_i^{t-j} \right) \cdot \hat{D}_{s,\pi}(s', s'') \end{aligned}$$

$$= \sum_{s' \in T^{s,\pi}} \left(\sum_{i=1}^{m} \sum_{j=0}^{n_i - 1} a_{ij}^{s'} \cdot \sum_{t=0}^{T-1} \frac{t!}{(t-j)!} \cdot \lambda_i^{t-j} \right) \cdot \hat{D}_{s,\pi}(s', s''). \tag{5.10}$$

Notice that for each j, we can obtain

$$\sum_{t=0}^{T-1} t^j \lambda_i^t = p_j(T) \lambda_i^T + C_j,$$

where $p_j(T)$ is a j-th order polynomial of T and C_j is a constant, by repeatedly taking derivatives and multiplying by λ_i starting with

$$\sum_{t=0}^{T-1} \lambda_i^t = \frac{1 - \lambda_i^T}{1 - \lambda_i}.$$

Then it follows from Eq. (5.9) that the limit of Eq. (5.10) as $T \to \infty$ is finite, and thus the following limit

$$\lim_{T \to \infty} E^{s,\pi} \left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^\pi \right] \cdot P^{s,\pi}(s_T \in R^\pi)$$

is finite.

Conversely, that the above limit as $T \to \infty$ is finite implies that for each s'', the limit of Eq. (5.10) as $T \to \infty$ is finite, which holds if and only if Eq. (5.9) holds. Then it follows that Eq. (5.8) holds, and thus

$$\lim_{T \to \infty} E^{s,\pi} \left[\gamma^{w_T} \middle| \, s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0. \qquad \square$$

Lemma 5.8. *Assume that Condition 2.1 (Finite Model) holds. Let* $\pi \in \Pi^{SR}$*. For all* $s \notin R^\pi$*,*

$$\limsup_{T \to \infty} E^{s,\pi} \left[\gamma^{w_T} \middle| \, s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_T \notin R^\pi) > 0,$$

if and only if

$$\lim_{T \to \infty} E^{s,\pi} \left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^\pi \right] \cdot P^{s,\pi}(s_T \in R^\pi) = +\infty.$$

Proof. Notice that

$$\begin{aligned} E^{s,\pi} \left[\gamma^{w_T} \middle| \, s_T \notin R^\pi \right] \cdot P^{s,\pi}(s_T \notin R^\pi) &= \sum_{s' \in T^{s,\pi}} E^{s,\pi} \left[\gamma^{w_T} \middle| \, s_T = s' \right] \cdot P^{s,\pi}(s_T = s') \\ &= \sum_{s' \in T^{s,\pi}} \hat{D}_{s,\pi}^T(s, s') \end{aligned}$$

and

$$\begin{aligned} &E^{s,\pi} \left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^\pi \right] \cdot P^{s,\pi}(s_T \in R^\pi) \\ &= \sum_{t=1}^{T} E^{s,\pi} \left[\gamma^{w_t} \middle| \, s_T \in R^\pi, s_{t-1} \notin R^\pi, s_t \in R^\pi \right] \cdot P^{s,\pi}(s_T \in R^\pi, s_{t-1} \notin R^\pi, s_t \in R^\pi) \end{aligned}$$

$$= \sum_{t=1}^{T} E^{s,\pi}\left[\gamma^{w_t} \middle| \, s_{t-1} \notin R^{\pi}, s_t \in R^{\pi}\right] \cdot P^{s,\pi}(s_{t-1} \notin R^{\pi}, s_t \in R^{\pi})$$

$$= \sum_{t=1}^{T} \sum_{s' \in T^{s,\pi}} \sum_{s'' \in R^{s,\pi}} E^{s,\pi}\left[\gamma^{w_t} \middle| \, s_{t-1} = s', s_t = s''\right] \cdot P^{s,\pi}(s_{t-1} = s', s_t = s'').$$

We have

$$\begin{aligned}
& E^{s,\pi}\left[\gamma^{w_t} \middle| \, s_{t-1} = s', s_t = s''\right] \cdot P^{s,\pi}(s_{t-1} = s', s_t = s'') \\
&= \sum_{\substack{h_t \in H_t \\ s_{t-1} = s' \\ s_t = s''}} P^{s,\pi}(h_t)\gamma^{w_t} \\
&= \sum_{\substack{h_{t-1} \in H_{t-1} \\ s_{t-1} = s'}} \sum_{a \in A_{s'}} P^{s,\pi}(h_{t-1}) P^{s,\pi}(s_t = s'', a_t = a | s_{t-1} = s')\gamma^{w_{t-1}}\gamma^{r(s',a,s'')} \\
&= \sum_{\substack{h_{t-1} \in H_{t-1} \\ s_{t-1} = s'}} P^{s,\pi}(h_{t-1})\gamma^{w_{t-1}} \cdot \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a)\gamma^{r(s',a,s'')}.
\end{aligned}$$

Let

$$K = \begin{cases} \gamma^{r^{+}_{\max}} & 0 < \gamma < 1 \\ \gamma^{r^{-}_{\min}} & \gamma > 1. \end{cases}$$

It then holds that

$$\begin{aligned}
& E^{s,\pi}\left[\gamma^{w_t} \middle| \, s_{t-1} = s', s_t = s''\right] \cdot P^{s,\pi}(s_{t-1} = s', s_t = s'') \\
&= \sum_{\substack{h_{t-1} \in H_{t-1} \\ s_{t-1} = s'}} P^{s,\pi}(h_{t-1})\gamma^{w_{t-1}} \cdot \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a)\gamma^{r(s',a,s'')} \\
&\geq \sum_{\substack{h_{t-1} \in H_{t-1} \\ s_{t-1} = s'}} P^{s,\pi}(h_{t-1})\gamma^{w_{t-1}} \cdot \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a)K \\
&= E^{s,\pi}\left[\gamma^{w_{t-1}} \middle| \, s_{t-1} = s'\right] \cdot P^{s,\pi}(s_{t-1} = s') \cdot \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a)K \\
&= \hat{D}^{t-1}_{s,\pi}(s,s')K \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a).
\end{aligned}$$

Therefore, it holds that

$$\begin{aligned}
& E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^{\pi}\right] \cdot P^{s,\pi}(s_T \in R^{\pi}) \\
&= \sum_{t=1}^{T} \sum_{s' \in T^{s,\pi}} \sum_{s'' \in R^{s,\pi}} E^{s,\pi}\left[\gamma^{w_t} \middle| \, s_{t-1} = s', s_t = s''\right] \cdot P^{s,\pi}(s_{t-1} = s', s_t = s'') \\
&\geq \sum_{t=1}^{T} \sum_{s' \in T^{s,\pi}} \sum_{s'' \in R^{s,\pi}} \hat{D}^{t-1}_{s,\pi}(s,s')K \sum_{a \in A_{s'}} \pi(s',a)P(s''|s',a)
\end{aligned}$$

$$= \sum_{t=1}^{T} \sum_{s' \in T^{s,\pi}} \hat{D}_{s,\pi}^{t-1}(s,s') K \sum_{a \in A_{s'}} \pi(s',a) \sum_{s'' \in R^{s,\pi}} P(s''|s',a)$$

$$= \sum_{t=1}^{T} \sum_{s' \in T^{s,\pi}} \hat{D}_{s,\pi}^{t-1}(s,s') K$$

$$= K \sum_{t=1}^{T} E^{s,\pi}\left[\gamma^{w_{t-1}} \middle| \, s_{t-1} \notin R^{\pi}\right] \cdot P^{s,\pi}(s_{t-1} \notin R^{\pi}).$$

Since

$$\limsup_{T \to \infty} E^{s,\pi}\left[\gamma^{w_T} \middle| \, s_T \notin R^{\pi}\right] \cdot P^{s,\pi}(s_T \notin R^{\pi}) > 0,$$

there exists $M > 0$ such that

$$\limsup_{T \to \infty} E^{s,\pi}\left[\gamma^{w_T} \middle| \, s_T \notin R^{\pi}\right] \cdot P^{s,\pi}(s_T \notin R^{\pi}) \geq M.$$

Therefore, it holds that for any $T_0 > 0$, there exists $T_1 > T_0$ such that

$$E^{s,\pi}\left[\gamma^{w_{T_1}} \middle| \, s_{T_1} \notin R^{\pi}\right] \cdot P^{s,\pi}(s_{T_1} \notin R^{\pi}) > \frac{M}{2}.$$

Then we have

$$\lim_{T \to \infty} E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^{\pi}\right] \cdot P^{s,\pi}(s_T \in R^{\pi}) \geq K \sum_{t=1}^{\infty} E^{s,\pi}\left[\gamma^{w_{t-1}} \middle| \, s_{t-1} \notin R^{\pi}\right] \cdot P^{s,\pi}(s_{t-1} \notin R^{\pi}),$$

and there are an infinite number of terms that are greater than $\frac{M}{2}$ in the above summation. Since each term is nonnegative, the above limit converges to positive infinity as $T \to \infty$.

The other direction holds since otherwise the limit

$$\lim_{T \to \infty} E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \middle| \, s_T \in R^{\pi}\right] \cdot P^{s,\pi}(s_T \in R^{\pi})$$

should be finite according to Lemma 5.7. □

We now can consider conditions that are sufficient for the values to exist. We need to discuss concave and convex exponential utility functions separately, since different conditions are needed.

5.3.1 Concave Exponential Utility Functions

If the utility function is concave, $0 < \gamma < 1$ and $\iota = -1$. Therefore, for all policies $\pi \in \Pi$, all states $s \in S$, and all $T \in \mathbb{N}$, the finite horizon value $v_{\exp,T}^{\pi}(s)$ is negative, and the infinite horizon value $v_{\exp}^{\pi}(s)$ is nonpositive if it exists.

We have seen that the value may not exist even if Condition 5.1 (One-Sided Finite Expected Rewards) holds. We therefore propose to use the following condition instead.

Condition 5.4 (One-Sided Finite Expected Concave Exponential Utilities).[1] For all policies $\pi \in \Pi$ and all states $s \in S$, at least one of $v^{+\pi}(s)$ and $v^{-\pi}_{\text{exp}}(s)$ is finite.

Condition 5.4 is stronger than Condition 5.1. They are related by the following theorem.

Theorem 5.9. *Assume that Condition 2.1 (Finite Model) holds. If $0 < \gamma < 1$, Condition 5.4 (One-Sided Finite Expected Concave Exponential Utilities) implies Condition 5.1 (One-Sided Finite Expected Rewards).*

To prove Theorem 5.9, we prove the following lemma for negative models. Since the negative part of an MDP is a negative model, that $v^{-\pi}_{\text{exp}}(s)$ is finite implies that $v^{-\pi}(s)$ is also finite according to Lemma 5.10(b). Therefore, Condition 5.4 implies Condition 5.1.

Lemma 5.10. *Assume that Condition 2.1 (Finite Model) and Condition 2.5 (Negative Model) hold.*

a. *If Condition 3.2 (Negative Model with Finite Exponential Utilities) holds for some $0 < \gamma < 1$ and some policy π, then it also holds for all γ' with $\gamma < \gamma' < 1$ and the same policy π.*

b. *If Condition 3.2 (Negative Model with Finite Exponential Utilities) holds for some $0 < \gamma < 1$ and some policy π, then Condition 2.6 (Negative Model with Finite Expected Rewards) holds for the same policy π.*

Proof. Obviously, $w_T \leq 0$ for all T.

a. When $0 < \gamma < 1$ and $w \leq 0$, the utility of w

$$U_{\exp(\gamma)}(w) = \iota\gamma^w = -\gamma^w$$

is nondecreasing with respect to γ. For all states $s \in S$, and all $\gamma' \in (\gamma, 1)$, if Condition 3.2 holds for γ and π, then

$$-\infty < v^{\pi}_{\exp(\gamma)}(s) = \lim_{T\to\infty} E^{s,\pi}\left[U_{\exp(\gamma)}(w_T)\right] \leq \lim_{T\to\infty} E^{s,\pi}\left[U_{\exp(\gamma')}(w_T)\right] = v^{\pi}_{\exp(\gamma')}(s).$$

Therefore, Condition 3.2 also holds for γ' and π.

[1] This phrase does not fully capture this condition. It is used just as a reminder of the condition.

b. It is sufficient to have, for all states $s \in S$,

$$\lim_{\gamma\to 1} U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma)}(s)\right) = v^{\pi}(s).$$

For all finite horizons $T \geq 1$, we have

$$U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma),T}(s)\right) = U^{-1}_{\exp(\gamma)}\left(E^{s,\pi}\left[U_{\exp(\gamma)}(w_T)\right]\right) = \log_{\gamma}\left(\iota E^{s,\pi}\left[\iota\gamma^{w_T}\right]\right) = \log_{\gamma} E^{s,\pi}\left[\gamma^{w_T}\right].$$

Therefore,

$$\begin{aligned}\lim_{\gamma\to 1} U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma),T}(s)\right) &= \lim_{\gamma\to 1}\log_{\gamma} E^{s,\pi}\left[\gamma^{w_T}\right] = \lim_{\gamma\to 1}\frac{\ln E^{s,\pi}\left[\gamma^{w_T}\right]}{\ln\gamma} = \lim_{\gamma\to 1}\frac{\dfrac{E^{s,\pi}\left[w_T\cdot\gamma^{w_T-1}\right]}{E^{s,\pi}\left[\gamma^{w_T}\right]}}{1/\gamma}\\ &= \lim_{\gamma\to 1}\frac{\gamma E^{s,\pi}\left[w_T\cdot\gamma^{w_T-1}\right]}{E^{s,\pi}\left[\gamma^{w_T}\right]} = E^{s,\pi}\left[w_T\right] = v^{\pi}_T(s),\end{aligned}$$

where we used l'Hôspital's rule. Consequently,

$$\begin{aligned}\lim_{\gamma\to 1} U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma)}(s)\right) &= \lim_{\gamma\to 1}\lim_{T\to\infty} U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma),T}(s)\right) = \lim_{T\to\infty}\lim_{\gamma\to 1} U^{-1}_{\exp(\gamma)}\left(v^{\pi}_{\exp(\gamma),T}(s)\right)\\ &= \lim_{T\to\infty} v^{\pi}_T(s) = v^{\pi}(s),\end{aligned}$$

where the order of limits can be changed since $v^{\pi}_{\exp(\gamma)}(s)$ is finite for all γ close to 1, which is implied by the premise and part (**a**). □

Condition 5.4 excludes the MDP shown in Figure 5.4(a), since it has both $v^{+\pi}(s^1) = v^{+\pi}(s^2) = +\infty$ and $v^{-\pi}_{\exp}(s^1) = v^{-\pi}_{\exp}(s^2) = -\infty$. It follows from Lemma 5.1 that $v^{+\pi}(s^3) = +\infty$, and thus $v^{+\pi}(s^1) = v^{+\pi}(s^2) = +\infty$. To verify that $v^{-\pi}_{\exp}(s^1) = v^{-\pi}_{\exp}(s^2) = -\infty$, we consider the matrix $D_{-\pi}$ that corresponds to the negative part of the model

$$D_{-\pi} = \begin{pmatrix} 0 & 1 & 1\\ 1 & 0 & \frac{1}{2}\\ 0 & 0 & 1\end{pmatrix}.$$

We can verify that

$$D^T_{-\pi} = \begin{pmatrix} \frac{1}{2}(1+(-1)^T) & \frac{1}{2}(1-(-1)^T) & \frac{1}{8}-\frac{1}{8}(-1)^T+\frac{3}{4}T\\ \frac{1}{2}(1-(-1)^T) & \frac{1}{2}(1+(-1)^T) & -\frac{1}{8}+\frac{1}{8}(-1)^T+\frac{3}{4}T\\ 0 & 0 & 1\end{pmatrix}.$$

The finite horizon values then are

$$v^{-\pi}_{\exp,T}(s^1) = -\frac{9}{8}+\frac{1}{8}(-1)^T-\frac{3}{4}T,$$

$$
\begin{aligned}
v^{-\pi}_{\exp,T}(s^2) &= -\frac{7}{8} - \frac{1}{8}(-1)^T - \frac{3}{4}T, \\
v^{-\pi}_{\exp,T}(s^3) &= -1.
\end{aligned}
$$

Therefore, $v^{-\pi}_{\exp}(s^1) = v^{-\pi}_{\exp}(s^2) = -\infty$. Thus this MDP does not satisfy Condition 5.4.

We now show that under Condition 5.4, the values under SR policies exist, as stated by the following theorem.

Theorem 5.11. *Assume that Condition 2.1 (Finite Model) and Condition 5.4 (One-Sided Finite Expected Concave Exponential Utilities) hold, and that $0 < \gamma < 1$. Let π be an SR policy. Then for all states $s \in S$, the value $v^{\pi}_{\exp}(s)$ exists.*

Proof. According to Condition 5.4 and Theorem 5.9, Condition 5.1 holds. Therefore, if $s \in R^\pi$, the results hold trivially according to Lemma 5.1.

Suppose $s \notin R^\pi$. Recall that the finite horizon values can be decomposed as

$$
\begin{aligned}
v^{\pi}_{\exp,T}(s) &= E^{s,\pi}[-\gamma^{w_T}] \\
-v^{\pi}_{\exp,T}(s) &= E^{s,\pi}[\gamma^{w_T}] \\
&= E^{s,\pi}[\gamma^{w_T} | s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\quad + E^{s,\pi}[\gamma^{w_{\tau(T)}} | s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad + E^{s,\pi}[\gamma^{w_T} - \gamma^{w_{\tau(T)}} | s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi). \qquad (5.11)
\end{aligned}
$$

If $v^{-\pi}_{\exp}(s)$ is finite, it also holds that $v^{-\pi}(s)$ is finite according to Lemma 5.10(b). Therefore,

$$
\begin{aligned}
-v^{-\pi}_{\exp,T}(s) &= E^{s,\pi}\left[\gamma^{w_T^-}\right] \\
&= E^{s,\pi}\left[\gamma^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma^{w_T^-} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&= E^{s,\pi}\left[\gamma^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma^{w_{\tau(T)}^-} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned}
$$

Since $v^{-\pi}_{\exp}(s)$ is finite and both terms in above are positive, the limit of the second term in above as $T \to \infty$ is also finite. It then follows from Lemma 5.7 that

$$
\lim_{T\to\infty} E^{s,\pi}\left[\gamma^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0.
$$

Since $0 < \gamma < 1$ and $w_T \geq w_T^-$, we have

$$
0 \leq E^{s,\pi}\left[\gamma^{w_T} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) \leq E^{s,\pi}\left[\gamma^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi),
$$

and thus

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma^{w_T} \,|\, s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0. \tag{5.12}$$

Lemma 5.7 implies that the limit

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma^{w_{\tau(T)}} \,|\, s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi)$$

exists and is finite. Therefore, the sum of the first two terms in Eq. (5.11) converges to finite values.

Since Condition 5.1 holds, we can distinguish the following cases according to Lemma 5.2.

1. Both $v^{+\pi}(s)$ and $v_{\exp}^{-\pi}(s)$ are finite. According to Lemma 5.3, the third term in Eq. (5.11) is zero. We have shown in above that the first two terms in Eq. (5.11) converge to finite values. Therefore, the value $v_{\exp}^{\pi}(s)$ is finite.

2. $v^{+\pi}(s)$ is finite and $v_{\exp}^{-\pi}(s) = -\infty$.

   (a) $v^{-\pi}(s) = -\infty$. In this case, the third term in Eq. (5.11) approaches positive infinity according to Lemma 5.3. Since the first two terms in Eq. (5.11) are also nonnegative, it holds that $v_{\exp}^{\pi}(s) = -\infty$.

   (b) $v^{-\pi}(s)$ is finite. In this case, the third term in Eq. (5.11) is zero according to Lemma 5.3 since the utility function approaches zero as $w \to \infty$. It holds that either

   $$\limsup_{T\to\infty} E^{s,\pi}\left[\gamma^{w_T} \,|\, s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) > 0$$

   or

   $$\lim_{T\to\infty} E^{s,\pi}\left[\gamma^{w_T} \,|\, s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0.$$

   In the first case, the second term in Eq. (5.11) approaches positive infinity according to Lemma 5.8 and thus the value $v_{\exp}^{\pi}(s) = -\infty$; in the second case, the second term in Eq. (5.11) approaches a finite value according to Lemma 5.7 and thus the value $v_{\exp}^{\pi}(s)$ is finite.

3. $v^{+\pi}(s) = \infty$ and $v_{\exp}^{-\pi}(s)$ is finite. We have shown in above that first two terms in Eq. (5.11) converge to finite values. According to Lemma 5.3, the third term of Eq. (5.11) converges to a finite number since the utility function approaches zero as $w \to \infty$. Therefore, the value $v_{\exp}^{\pi}(s)$ exists and is finite. □

However, it is still an open problem whether there exists an SD-optimal policy for MDPs with both positive and negative rewards under the $\mathsf{MEU}_{\exp}$ objective. Therefore, the above theorem is not sufficient to show that the optimal values exist. I nevertheless conjecture that the above result also holds for all policies, therefore the optimal values exist.

Conjecture 5.1. *Assume that Condition 2.1 (Finite Model) and Condition 5.4 (One-Sided Finite Expected Concave Exponential Utilities) hold, and that* $0 < \gamma < 1$. *Let* π *be an HR policy. Then for all states* $s \in S$, *the values* $v^{\pi}_{\text{exp}}(s)$ *exist. Consequently, for all states* $s \in S$, *the optimal values* $v^{*}_{\text{exp}}(s)$ *exist.*

For the optimal values to be finite, we use the following condition.

Condition 5.5 (Finite Negative Part Expected Exponential Utilities)**.** There exists a policy π such that for all states $s \in S$, the value $v^{-\pi}_{\text{exp}}(s)$ is finite.

Theorem 5.12. *Assume that Condition 2.1 (Finite Model), Condition 5.4 (One-Sided Finite Expected Concave Exponential Utilities), and Condition 5.5 (Finite Negative Part Expected Exponential Utilities) hold, and that Conjecture 5.1 holds. Then for all states* $s \in S$, *the optimal value* $v^{*}_{\text{exp}}(s)$ *is finite.*

Proof. Let π be the policy for which Condition 5.5 (Finite Negative Part Expected Exponential Utilities) holds. For all states $s \in S$, we have

$$v^{\pi}_{\text{exp},T}(s) = E^{s,\pi}\left[U_{\text{exp}}\left(\sum_{t=0}^{T-1} r_t\right)\right] \geq E^{s,\pi}\left[U_{\text{exp}}\left(\sum_{t=0}^{T-1} r_t^{-}\right)\right] = v^{-\pi}_{\text{exp},T}(s).$$

Taking the limit as T approaches infinity shows that $v^{*}_{\text{exp}}(s) \geq v^{\pi}_{\text{exp}}(s) \geq v^{-\pi}_{\text{exp}}(s) > -\infty$. □

5.3.2 Convex Exponential Utility Functions

The results and proofs for convex exponential utility functions are parallel to those for concave exponential utility functions. Therefore, we list these results, and only provide proofs for results that are not completely symmetric to the case of concave exponential utility functions.

For convex exponential utility functions, $\gamma > 1$ and $\iota = 1$. Therefore, for all policies $\pi \in \Pi$, all states $s \in S$, and all $T \in \mathbb{N}$, the finite horizon value $v^{\pi}_{\text{exp},T}(s)$ is positive, and the infinite horizon value $v^{\pi}_{\text{exp}}(s)$ is nonnegative if it exists.

Condition 5.6 (One-Sided Finite Expected Convex Exponential Utilities)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, at least one of $v^{+\pi}_{\text{exp}}(s)$ and $v^{-\pi}(s)$ is finite.

Condition 5.6 is stronger than Condition 5.1 (One-Sided Finite Expected Rewards). They are related by the following theorem.

Theorem 5.13. *Assume that Condition 2.1 (Finite Model) holds. If $\gamma > 1$, Condition 5.6 (One-Sided Finite Expected Convex Exponential Utilities) implies Condition 5.1 (One-Sided Finite Expected Rewards).*

Similar to the case of concave exponential utility functions, we need the following lemma for the proof of Theorem 5.13.

Lemma 5.14. *Assume that Condition 2.1 (Finite Model) and Condition 2.7 (Positive Model) hold.*

a. *If Condition 3.4 (Positive Model with Finite Exponential Utilities) holds for some $\gamma > 1$, then it also holds for all γ' with $\gamma > \gamma' > 1$.*

b. *If Condition 3.4 (Positive Model with Finite Exponential Utilities) holds for some $\gamma > 1$, then Condition 2.8 (Positive Model with Finite Expected Rewards) holds.*

Under Condition 5.6, the values under SR policies exist, as stated by the following theorem.

Theorem 5.15. *Assume that Condition 2.1 (Finite Model) and Condition 5.6 (One-Sided Finite Expected Convex Exponential Utilities) hold, and that $\gamma > 1$. Let π be an SR policy. Then for all states $s \in S$, the value $v^{\pi}_{\text{exp}}(s)$ exists.*

However, this theorem is not sufficient to show that the optimal values exist. I conjecture that the result also holds for all policies, and therefore the optimal values exist.

Conjecture 5.2. *Assume that Condition 2.1 (Finite Model) and Condition 5.6 (One-Sided Finite Expected Convex Exponential Utilities) hold, and that $\gamma > 1$. Let π be an HR policy. Then for all states $s \in S$, the value $v^{\pi}_{\text{exp}}(s)$ exists. Therefore, for all states $s \in S$, the optimal value $v^{*}_{\text{exp}}(s)$ exists.*

Conjecture 5.2 (with risk parameter γ) in fact is equivalent to Conjecture 5.1 (with risk parameter γ^{-1}). We can replace $r(s, a, s')$ with $r'(s, a, s') = -r(s, a, s')$ and thus

$r^+(s,a,s')$ with $r'^+(s,a,s') = -r^-(s,a,s')$ and $r^-(s,a,s')$ with $r'^-(s,a,s') = -r^+(s,a,s')$. Consequently, Conjecture 5.2 is reduced to Conjecture 5.1.

For the optimal values to be finite, it is necessary to have the following condition, which obviously implies Condition 5.6.

Condition 5.7 (Finite Positive Part Expected Exponential Utilities)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, $v^{+\pi}_{\exp}(s)$ is finite.

Theorem 5.16. *Assume that Condition 2.1 (Finite Model), Condition 5.6 (One-Sided Finite Expected Convex Exponential Utilities), and Conjecture 5.2 hold. Then for all states $s \in S$, the optimal value $v^*_{\exp}(s)$ is finite.*

Proof. Condition 5.7 (Finite Positive Part Expected Exponential Utilities) implies that for all states $s \in S$, $v^{+*}_{\exp}(s)$ is finite. Furthermore, for all policies $\pi \in \Pi$ and all states $s \in S$,

$$v^\pi_{\exp,T}(s) = E^{s,\pi}\left[U_{\exp}\left(\sum_{t=0}^{T-1} r_t\right)\right] \le E^{s,\pi}\left[U_{\exp}\left(\sum_{t=1}^{T-1} r^+_t\right)\right] = v^{+\pi}_{\exp,T}(s).$$

Taking the limit as T approaches infinity shows that $v^\pi_{\exp}(s) \le v^{+\pi}_{\exp}(s) < \infty$. Therefore, $v^*_{\exp}(s) \le v^{+*}_{\exp}(s) < \infty$. □

5.4 General Risk-Sensitive Utility Functions

We consider general risk-sensitive utility functions in this section. The conditions to ensure the existence and finiteness of values can be classified into two classes. The first class includes conditions on the utility functions, and the second class includes conditions on the MDP models. The conditions in each class range from the most general ones to the most restrictive ones. Each set of conditions to be presented below is a combination of conditions from each class.

The most general condition for utility functions is Condition 4.1 (Nondecreasing Utility Function), and the most general condition for MDPs is Condition 5.1 (One-Sided Finite Expected Rewards).

5.4.1 Bounded Utility Functions

We now consider the case where the utility functions are bounded, which is the most restrictive condition for utility functions. In this case, $v^\pi_{U,T}(s)$ is bounded as T approaches infinity but the limit might not exist since the values can oscillate. Theorem 5.17 provides a condition that guarantees the existence of values for stationary policies.

Condition 5.8 (Bounded Utility Function)**.** The utility function is bounded, that is, $\lim_{w \to -\infty} U(w) = U^-$ and $\lim_{w \to \infty} U(w) = U^+$ where $U^+ \neq U^-$ are finite.[2]

Theorem 5.17. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.8 (Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an SR policy. Then for all states $s \in S$, $v_U^\pi(s)$ exists and is bounded.*

Proof. Following the discussion in Section 5.2, we decompose the finite horizon values as

$$\begin{aligned}
v_{U,T}^\pi(s) &= E^{s,\pi}\left[U(w_T)\right] \\
&= E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\quad + E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad + E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned} \tag{5.13}$$

Now we consider the first term. Since by definition,

$$\lim_{T \to \infty} P^{s,\pi}(s_T \notin R^\pi) = 0,$$

and

$$\left| E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \right| \leq \max\left(\left|U^+\right|, \left|U^-\right|\right),$$

we have

$$\lim_{T \to \infty} E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0.$$

For the second term, we have

$$\begin{aligned}
&E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad = \sum_{t=1}^{T} E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi) \\
&\quad = \sum_{t=1}^{T} E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi).
\end{aligned}$$

Therefore,

$$\left| E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \right|$$

[2] If $U^+ = U^-$, it follows that $U(w) \equiv U$, which is a boring utility function.

$$
\begin{aligned}
&\le \sum_{t=1}^{T} E^{s,\pi}\left[\left|U(w_t)\right| \middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\
&\le \sum_{t=1}^{T} \max\left(\left|U^+\right|, \left|U^-\right|\right) \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\
&= \max\left(\left|U^+\right|, \left|U^-\right|\right) \cdot \sum_{t=1}^{T} P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\
&= \max\left(\left|U^+\right|, \left|U^-\right|\right) \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned}
$$

Since

$$\lim_{T\to\infty} P^{s,\pi}(s_T \in R^\pi) = 1,$$

it follows that the limit

$$\lim_{T\to\infty} E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

converges to a finite value.

We also know from Lemma 5.3 that the third term converges to a finite value as $T \to \infty$. Therefore, $v_U^\pi(s)$ exists and is finite. □

We have seen in Chapter 4 that the optimal policy can be history-dependent for risk-sensitive planning with general utility functions, so the above theorem is not sufficient for the optimal values to exist. I conjecture that the above result also holds for all policies, therefore the optimal values exist. Since the utility function is bounded, the optimal values are also finite, provided the conjecture holds.

Conjecture 5.3. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.8 (Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an HR policy. Then for all states $s \in S$, $v_U^\pi(s)$ exists and is bounded. Therefore, for all states $s \in S$, $v_U^*(s)$ exists and is finite.*

5.4.2 Linearly Bounded Utility Functions

Next, we consider the case where the utility functions are bounded by linear functions.

Condition 5.9 (Linearly Bounded Utility Function)**.** The utility function is linearly bounded, that is, $U(w) = O(w)$ as $w \to \infty$ and as $w \to -\infty$.

The utility function $U(w) = O(w)$ implies that there exists positive numbers C and D such that $U(w) \leq Cw + D$ when $w > 0$ and $U(w) \geq Cw - D$ when $w < 0$.

The following theorem shows that the values exist for SR policies under conditions that are, in part, similar to those for the MER objective.

Theorem 5.18. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.9 (Linearly Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an SR policy. Then for all states $s \in S$, $v_U^\pi(s)$ exists.*

Proof. Following the discussion in Section 5.2, we decompose the finite horizon values as

$$\begin{aligned}
v_{U,T}^\pi(s) &= E^{s,\pi}\left[U(w_T)\right] \\
&= E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\quad + E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad + E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi). \qquad (5.14)
\end{aligned}$$

Consider the first term in Eq. (5.14). By definition, we have

$$\lim_{T\to\infty} P^{s,\pi}(s_T \notin R^\pi) = 0.$$

We also have

$$Cw_T^- - D \leq U(w_T^-) \leq U(w_T) \leq U(w_T^+) \leq Cw_T^+ + D,$$

and thus

$$\left|E^{s,\pi}[U(w_T)|s_T \notin R^\pi]\right| \leq C \max\left(E^{s,\pi}[w_T^+|s_T \notin R^\pi], -E^{s,\pi}[w_T^-|s_T \notin R^\pi]\right) + D.$$

Therefore, we have

$$\begin{aligned}
&\left|E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi)\right| \\
&\qquad \leq C \max\left(E^{s,\pi}[w_T^+|s_T \notin R^\pi], -E^{s,\pi}[w_T^-|s_T \notin R^\pi]\right) \cdot P^{s,\pi}(s_T \notin R^\pi) + D \cdot P^{s,\pi}(s_T \notin R^\pi).
\end{aligned}$$

Since for linear utility functions, it follows from Lemma 5.4 that

$$\begin{aligned}
&\lim_{T\to\infty} E^{s,\pi}[w_T^+|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0, \\
&\lim_{T\to\infty} E^{s,\pi}[w_T^-|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0,
\end{aligned}$$

and we also have

$$\lim_{T\to\infty} P^{s,\pi}(s_T \notin R^\pi) = 0,$$

it follows that

$$\lim_{T\to\infty} E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0.$$

For the second term in Eq. (5.14), we have

$$\begin{aligned}
&E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad = \sum_{t=1}^{T} E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi) \\
&\quad = \sum_{t=1}^{T} E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi). \qquad (5.15)
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\left|E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)\right| \\
&\le \sum_{t=1}^{T} E^{s,\pi}\left[\left|U(w_t)\right|\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\
&\le \sum_{t=1}^{T} \Big(C \max\big(E^{s,\pi}[w_t^+|s_t \in R^\pi, s_{t-1} \notin R^\pi], \\
&\qquad\qquad - E^{s,\pi}[w_t^-|s_t \in R^\pi, s_{t-1} \notin R^\pi]\big) + D\Big) \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \in R^\pi) \\
&= C \max\Bigg(\sum_{t=1}^{T} E^{s,\pi}[w_t^+|s_t \in R^\pi, s_{t-1} \notin R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi), \\
&\qquad\qquad - \sum_{t=1}^{T} E^{s,\pi}[w_t^-|s_t \in R^\pi, s_{t-1} \notin R^\pi] \cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi)\Bigg) \\
&\qquad + D \cdot \sum_{t=1}^{T} P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\
&= C \max\Big(E^{s,\pi}[w_{\tau(T)}^+|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi), -E^{s,\pi}[w_{\tau(T)}^-|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)\Big) \\
&\qquad + D \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned}$$

Since

$$\lim_{T\to\infty} P^{s,\pi}(s_T \in R^\pi) = 1,$$

and the limits

$$\lim_{T\to\infty} E^{s,\pi}[w_{\tau(T)}^+|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \quad \text{and} \quad \lim_{T\to\infty} E^{s,\pi}[w_{\tau(T)}^-|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

exist and are finite according to Lemma 5.4, it follows that the summation in Eq. (5.15) converges absolutely as $T \to \infty$. Therefore, the limit

$$\lim_{T\to\infty} E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

exists and is finite.

We also know from Lemma 5.3 that the third term in Eq. (5.14) converges. Therefore, $v_U^\pi(s)$ exists. To be more specific, we have

- if $v^\pi(s)$ is finite, then the third term in Eq. (5.14) converges to zero and $v_U^\pi(s)$ is finite;
- if $v^\pi(s) = \infty$ and the utility function is bounded from above, then the third term in Eq. (5.14) converges to a finite number and $v_U^\pi(s)$ is also finite;
- if $v^\pi(s) = \infty$ and the utility function is unbounded from above, then the third term in Eq. (5.14) converges to positive infinity and $v_U^\pi(s) = \infty$;
- if $v^\pi(s) = -\infty$ and the utility function is bounded from below, then the third term in Eq. (5.14) converges to a finite number and $v_U^\pi(s)$ is also finite; and
- if $v^\pi(s) = -\infty$ and the utility function is unbounded from below, then the third term in Eq. (5.14) converges to negative infinity and $v_U^\pi(s) = -\infty$. □

Again, the above theorem is not sufficient for the optimal values to exist. I conjecture the following result holds.

Conjecture 5.4. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.9 (Linearly Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.1 (One-Sided Finite Expected Rewards) hold. Let π be an HR policy. Then for all states $s \in S$, $v_U^\pi(s)$ exists. Therefore, for all states $s \in S$, $v_U^*(s)$ exists.*

Theorem 5.19. *Assume that Condition 2.1 (Finite Model), Condition 5.2 (Finite Positive-Part Expected Rewards), Condition 5.3 (Finite Negative-Part Expected Rewards), Condition 5.9 (Linearly Bounded Utility Function), and Conjecture 5.4 hold. Then the optimal values are finite.*

Proof. For any given policy π, we have for all states s and all $T \in \mathbb{N}$,

$$v_{U,T}^\pi(s) \leq v_{U,T}^{+\pi}(s) \leq C v_T^{+\pi}(s) + D.$$

Therefore, it follows from Condition 5.2 and Condition 5.9 that

$$\limsup_{T\to\infty} v_{U,T}^\pi(s) \leq \lim_{T\to\infty} \left(C v_T^{+\pi}(s) + D \right) \leq C v^{+\pi}(s) + D \leq C v^{+*}(s) + D.$$

Since π is arbitrary, it holds that

$$v_U^*(s) \le Cv^{+*}(s) + D < \infty.$$

Suppose Condition 5.3 holds for policy π'. We have for all states $s \in S$,

$$v_{U,T}^{\pi'}(s) \ge v_{U,T}^{-\pi'}(s) \ge Cv_T^{-\pi'}(s) - D.$$

Therefore, it holds that

$$v_U^*(s) \ge \liminf_{T\to\infty} v_{U,T}^{\pi'}(s) \ge \lim_{T\to\infty}\left(Cv_T^{-\pi'}(s) - D\right) = Cv^{-\pi'}(s) - D > -\infty.$$

That is, the optimal values are finite. □

5.4.3 Exponentially Bounded Utility Functions

We consider exponentially bounded utility functions in this section.

Condition 5.10 (Exponentially Bounded Utility Function)**.** The utility function is exponentially bounded, that is, $U(w) = O(\gamma_+^w)$ as $w \to \infty$ where $\gamma_+ > 1$, and $U(w) = O(\gamma_-^w)$ as $w \to -\infty$ where $0 < \gamma_- < 1$.

Condition 5.10 is equivalent to the following condition: there exists $C > 0$ and $D > 0$ such that $U(w) \le C\gamma_+^w + D$ if $w > 0$ and $U(w) \ge -C\gamma_-^w - D$ if $w < 0$.

For such utility functions, the following condition is sufficient for the existence and finiteness of values for SR policies, as stated in Theorem 5.20.

Condition 5.11 (Finite Expected Exponential Utilities)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, $v_{\exp(\gamma_+)}^{+\pi}(s)$ and $v_{\exp(\gamma_-)}^{-\pi}(s)$ are finite, where $\gamma_+ > 1$ and $0 < \gamma_- < 1$ are the risk parameters from Condition 5.10.

Theorem 5.20. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.10 (Exponentially Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.11 (Finite Expected Exponential Utilities) hold. Then for all SR policies $\pi \in \Pi^{SR}$ and all states $s \in S$, the value $v_U^\pi(s)$ exists.*

Proof. Following the discussion in Section 5.2, we decompose the finite horizon values as

$$v_{U,T}^\pi(s) = E^{s,\pi}\left[U(w_T)\right]$$

$$\begin{aligned}
&= E^{s,\pi}[U(w_T)|s_T \notin R^\pi] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\quad + E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&\quad + E^{s,\pi}[U(w_T) - U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi). \qquad (5.16)
\end{aligned}$$

Since $v^{+\pi}_{\exp(\gamma_+)}(s)$ is finite, it also hold that $v^{+\pi}(s)$ is finite according to Theorem 5.13. Therefore,

$$\begin{aligned}
v^{+\pi}_{\exp(\gamma_+),T}(s) &= E^{s,\pi}\left[\gamma_+^{w_T^+}\right] \\
&= E^{s,\pi}\left[\gamma_+^{w_T^+} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma_+^{w_T^+} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&= E^{s,\pi}\left[\gamma_+^{w_T^+} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma_+^{w_{\tau(T)}^+} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned}$$

It then follows from Lemma 5.7 that

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma_+^{w_T^+} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0$$

and the following limit exists and is finite

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma_+^{w_{\tau(T)}^+} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi) = v^{+\pi}_{\exp(\gamma_+)}(s). \qquad (5.17)$$

Since $U(w) \le C\gamma_+^w + D$ for $w \ge 0$ and $w_T \le w_T^+$, we have

$$\begin{aligned}
E^{s,\pi}\left[U(w_T)\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) &\le E^{s,\pi}\left[U(w_T^+)\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\le E^{s,\pi}\left[C\gamma_+^{w_T^+} + D\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi).
\end{aligned}$$

and thus

$$\begin{aligned}
&\limsup_{T\to\infty} E^{s,\pi}\left[U(w_T)\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&\le \lim_{T\to\infty} E^{s,\pi}\left[C\gamma_+^{w_T^+} + D\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) \\
&= C \lim_{T\to\infty} E^{s,\pi}\left[\gamma_+^{w_T^+}\middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + D \lim_{T\to\infty} P^{s,\pi}(s_T \notin R^\pi) = 0. \qquad (5.18)
\end{aligned}$$

Similarly, since $v^{-\pi}_{\exp(\gamma_-)}(s)$ is finite, it also hold that $v^{-\pi}(s)$ is finite according to Theorem 5.9. Therefore,

$$\begin{aligned}
v^{-\pi}_{\exp(\gamma_-),T}(s) &= E^{s,\pi}\left[\gamma_-^{w_T^-}\right] \\
&= E^{s,\pi}\left[\gamma_-^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma_-^{w_T^-} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi) \\
&= E^{s,\pi}\left[\gamma_-^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) + E^{s,\pi}\left[\gamma_-^{w_{\tau(T)}^-} \middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi).
\end{aligned}$$

It then follows from Lemma 5.7 that

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma_-^{w_T^-} \middle| s_T \notin R^\pi\right] \cdot P^{s,\pi}(s_T \notin R^\pi) = 0$$

and the following limit exists and is finite

$$\lim_{T\to\infty} E^{s,\pi}\left[\gamma_-^{w^-_{\tau(T)}}\middle| s_T \in R^\pi\right]\cdot P^{s,\pi}(s_T \in R^\pi) = v^{-\pi}_{\exp(\gamma_-)}(s). \tag{5.19}$$

Since $U(w) \geq -C\gamma_-^w - D$ for $w \leq 0$ and $w_T \geq w_T^-$, we have

$$\begin{aligned} E^{s,\pi}\left[U(w_T)\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) &\geq E^{s,\pi}\left[U(w_T^-)\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) \\ &\geq E^{s,\pi}\left[-C\gamma_-^{w_T^-} - D\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi). \end{aligned}$$

and thus

$$\begin{aligned} &\liminf_{T\to\infty} E^{s,\pi}\left[U(w_T)\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) \\ &\geq \lim_{T\to\infty} E^{s,\pi}\left[-C\gamma_-^{w_T^-} - D\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) \\ &= -C\lim_{T\to\infty} E^{s,\pi}\left[\gamma_-^{w_T^-}\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) - D\lim_{T\to\infty} P^{s,\pi}(s_T \notin R^\pi) = 0. \end{aligned} \tag{5.20}$$

Then it follows from Eq. (5.18) and Eq. (5.20) that the first term in Eq. (5.16) is

$$\lim_{T\to\infty} E^{s,\pi}\left[U(w_T)\middle| s_T \notin R^\pi\right]\cdot P^{s,\pi}(s_T \notin R^\pi) = 0.$$

To consider the second term in Eq. (5.16), we use the following decomposition,

$$\begin{aligned} &E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi]\cdot P^{s,\pi}(s_T \in R^\pi) \\ &\quad = \sum_{t=1}^T E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi]\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi, s_T \in R^\pi) \\ &\quad = \sum_{t=1}^T E^{s,\pi}[U(w_t)|s_t \in R^\pi, s_{t-1} \notin R^\pi]\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi). \end{aligned}$$

Therefore,

$$\begin{aligned} &\left|E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi]\cdot P^{s,\pi}(s_T \in R^\pi)\right| \\ &\leq \sum_{t=1}^T E^{s,\pi}\left[\left|U(w_t)\right|\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right]\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \\ &\leq \sum_{t=1}^T \Bigg(C\max\Big(E^{s,\pi}\left[\gamma_+^{w_t^+}\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right], \\ &\qquad\qquad E^{s,\pi}\left[\gamma_-^{w_t^-}\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right]\Big) + D\Bigg)\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \in R^\pi) \\ &= C\max\Bigg(\sum_{t=1}^T E^{s,\pi}\left[\gamma_+^{w_t^+}\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right]\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi), \\ &\qquad\qquad \sum_{t=1}^T E^{s,\pi}\left[\gamma_-^{w_t^-}\middle| s_t \in R^\pi, s_{t-1} \notin R^\pi\right]\cdot P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi)\Bigg) \end{aligned}$$

$$+ D \cdot \sum_{t=1}^{T} P^{s,\pi}(s_t \in R^\pi, s_{t-1} \notin R^\pi) \qquad \triangleright \quad \text{both } \gamma_+^{w_t^+} \text{ and } \gamma_-^{w_t^-} \text{ are positive}$$

$$= C \max\left(E^{s,\pi}\left[\gamma_+^{w_{\tau(T)}^+}\middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi), E^{s,\pi}\left[\gamma_-^{w_{\tau(T)}^-}\middle| s_T \in R^\pi\right] \cdot P^{s,\pi}(s_T \in R^\pi)\right)$$
$$+ D \cdot P^{s,\pi}(s_T \in R^\pi). \tag{5.21}$$

Since

$$\lim_{T\to\infty} P^{s,\pi}(s_T \in R^\pi) = 1,$$

it follows from Eq. (5.17), Eq. (5.19), and Eq. (5.21) that the quantity

$$E^{s,\pi}[U(w_{\tau(T)})|s_T \in R^\pi] \cdot P^{s,\pi}(s_T \in R^\pi)$$

converges absolutely as $T \to \infty$, and therefore the second term in Eq. (5.16) converges as $T \to \infty$.

According to Condition 5.11 (Finite Expected Exponential Utilities), Theorem 5.9, and Theorem 5.13, Condition 5.1 (One-Sided Finite Expected Rewards) holds. Then according to Lemma 5.3, the third term in Eq. (5.16) is zero.

Therefore, the value $v_U^\pi(s)$ is finite. □

Condition 5.11, however, is too restrictive, and may exclude interesting problems with improper policies, for example. Alternatively, an analogy to the case of linearly bounded utility functions suggests that we consider the following weaker condition.

Condition 5.12 (One-Sided Finite Expected Exponential Utilities)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, at least one of $v_{\exp(\gamma_+)}^{+\pi}(s)$ and $v_{\exp(\gamma_-)}^{-\pi}(s)$ is finite, where $\gamma_+ > 1$ and $0 < \gamma_- < 1$ are the risk parameters from Condition 5.10.

I conjecture that the values for all policies and the optimal values exist under this condition. However, it is even unclear how to show that the result holds for stationary policies.

Conjecture 5.5. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 5.10 (Exponentially Bounded Utility Function), Condition 2.1 (Finite Model), and Condition 5.12 (One-Sided Finite Expected Exponential Utilities) hold. Then for all policies $\pi \in \Pi$ and all states $s \in S$, the value $v_U^\pi(s)$ exists. Therefore, for all states $s \in S$, the optimal value $v_U^*(s)$ exists.*

5.4.4 Bounded Total Rewards and Arbitrary Utility Functions

We next consider the case where the total reward is bounded from below or above. We use $H_T^{s,\pi}$ to denote the set of trajectories with finite horizon T starting from $s \in S$ under policy $\pi \in \Pi$. We define $w(h_T) = \sum_{t=0}^{T-1} r_t$ for $h_T \in H_T^{s,\pi}$, where r_t is the t-th reward along the trajectory h_T. We also define the maximum finite horizon total reward as

$$v^{\pi}_{\max,T}(s) = \max_{h_T \in H_T^{s,\pi}} w(h_T)$$

and the minimum finite horizon total reward as

$$v^{\pi}_{\min,T}(s) = \min_{h_T \in H_T^{s,\pi}} w(h_T).$$

The maximum total reward (over an infinite horizon) then is

$$v^{\pi}_{\max}(s) = \lim_{T\to\infty} v^{\pi}_{\max,T}(s)$$

and the minimum total reward is

$$v^{\pi}_{\min}(s) = \lim_{T\to\infty} v^{\pi}_{\min,T}(s).$$

Notice that $v^{+\pi}_{\max}(s)$ and $v^{-\pi}_{\min}(s)$ are always well-defined since the maximum and minimum finite horizon values are monotonic in T. We therefore consider the following condition.

Condition 5.13 (One-Sided Finite Total Reward)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, at least one of $v^{+\pi}_{\max}(s)$ and $v^{-\pi}_{\min}(s)$ is finite.

Theorem 5.21. *Assume that Condition 2.1 (Finite Model) holds. Condition 5.13 (One-Sided Finite Total Reward) implies Condition 5.1 (One-Sided Finite Expected Rewards).*

Proof. We show that $v^{+\pi}_{\max}(s)$ is finite implies that $v^{+\pi}(s)$ is finite. For all finite horizon T, it holds that

$$v^{+\pi}_{\max,T}(s) \geq v^{+\pi}_T(s).$$

Therefore,

$$v^{+\pi}_{\max}(s) = \lim_{T\to\infty} v^{+\pi}_{\max,T}(s) \geq \lim_{T\to\infty} v^{+\pi}_T(s) = v^{+\pi}(s).$$

Similarly, it holds that $v^{-\pi}_{\min}(s)$ is finite implies that $v^{-\pi}(s)$ is finite. Therefore the result holds. □

Lemma 5.22. *Assume that Condition 2.1 (Finite Model) holds.*

a. *If $v^{+\pi}_{\max}(s)$ is finite, then for all valid transitions (s, a, s') where $r(s, a, s') > 0$, it holds that for all $t \geq 1$, $P^{s',\pi}(s_t = s) = 0$.*

b. *If $v^{-\pi}_{\min}(s)$ is finite, then for all valid transitions (s, a, s') where $r(s, a, s') < 0$, it holds that for all $t \geq 1$, $P^{s',\pi}(s_t = s) = 0$.*

Proof. We prove part (a), and part (b) is similar. Suppose otherwise: there exists $t_0 \geq 1$ and $P^{s',\pi}(s_{t_0} = s) > 0$. Then for all integer $n \geq 1$, the probability of visiting s for n times is positive, and thus receive a total positive part reward of at least $nr(s, a, s')$, which is greater than $v^{+\pi}_{\max}(s)$ if n is sufficiently large. Therefore the result holds. □

Theorem 5.23. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 2.1 (Finite Model), and Condition 5.13 (One-Sided Finite Total Reward) hold. Then for all policies $\pi \in \Pi$ and all states $s \in S$, the value $v^{\pi}_{U}(s)$ exists.*

Proof. We consider the following different cases.

1. Suppose both $v^{+\pi}_{\max}(s)$ and $v^{-\pi}_{\min}(s)$ are finite. In this case, it holds that

$$v^{-\pi}_{\min}(s) \leq v^{-\pi}_{\min,T}(s) \leq v^{\pi}_{\min,T}(s) \leq w_T \leq v^{\pi}_{\max,T}(s) \leq v^{+\pi}_{\max,T}(s) \leq v^{+\pi}_{\max}(s).$$

   According to Lemma 5.22, only transitions with zero rewards can be taken more than once. Since there are only a finite number of non-zero rewards, the total rewards for all trajectories are bounded and there are only a finite number of possible total rewards. Therefore, the risk-sensitive value for a state is an expectation over a finite set, and thus always exists.

2. Suppose $v^{+\pi}_{\max}(s)$ is finite and $v^{-\pi}_{\min}(s) = -\infty$. It follows from Lemma 5.22 that transitions with positive rewards can only be taken at most once, or equivalently, there are only a finite number of transitions with positive rewards in any trajectory under π. Consider the augmented model as defined in Section 4.3.1. Recall that we define $\langle r|_U\rangle(\langle s\rangle, a, \langle s\rangle) = U(w') - U(w)$, where $\langle s\rangle = (s, w)$ and $\langle s\rangle' = (s', w')$ where $w, w' \in W$, and the transition $(\langle s\rangle, a, \langle s\rangle')$ is possible if and only if $w' = w + r(s, a, s')$ and $P(s'|s, a) > 0$. Since U is monotonically nondecreasing, a transition in the original model with $r(s, a, s') > 0$ corresponds to a set of transitions in the augmented model with $\langle r|_U\rangle(\langle s\rangle, a, \langle s\rangle') \geq 0$ where $\langle s\rangle = (s, w)$ and $\langle s\rangle' = (s', w + r(s, a, s'))$ for all $w \in W$ and vice versa. Since there are only a finite number of transitions with

positive rewards in any trajectory under π in the original model and the original model and the augmented model induce the same random process of world states and actions under π and $\Psi(\pi)$, starting from s and $\langle s \rangle = (s, w)$ where $w \in W$, respectively, there are also only a finite number of transitions with positive rewards in any trajectory under $\Psi(\pi)$ in the augmented model. It then follows that $\langle v|_U \rangle^{+\Psi(\pi)}_{\max}(s, w)$ is also finite for all $w \in W$. That is, the augmented model is essentially negative under π (Hinderer, 1970, see below), and therefore $\langle v|_U \rangle^{\Psi(\pi)}(s, w)$ exists for all $w \in W$ (may be infinity). Then it follows from Theorem 4.11 that $v^{\pi}_U(s) = v^{\Phi_0(\Psi(\pi))}_U(s) = \langle v|_U \rangle^{\Psi(\pi)}(s, 0) + U(0)$ exists.

3. Suppose $v^{+\pi}_{\max}(s) = \infty$ and $v^{-\pi}_{\min}(s)$ is finite. This case is similar to the previous case. □

However, for the optimal values to be finite, we need the following pair of stronger conditions.

Condition 5.14 (Finite Total Positive Part Reward)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the values $v^{+\pi}_{\max}(s)$ are finite.

Condition 5.15 (Finite Total Negative Part Reward)**.** There exists a policy $\pi \in \Pi$ such that for all states $s \in S$, it holds that the values $v^{-\pi}_{\min}(s)$ are finite.

MDPs satisfying Condition 5.14 are also referred to as essentially negative models in the literature (Hinderer, 1970), since all, except for a finite number, of rewards along a trajectory are negative (or zero). Similarly, MDPs satisfying the following condition, which is stronger than Condition 5.15, are known as essentially positive models (Hinderer, 1970).

Condition 5.16 (All Finite Total Negative Part Reward)**.** For all policies $\pi \in \Pi$ and all states $s \in S$, the values $v^{-\pi}_{\min}(s)$ are finite.

Condition 5.14 and Condition 5.15 (even Condition 5.16) are, for example, satisfied for acyclic MDPs if plan execution ends in absorbing states but are satisfied for some cyclic MDPs as well.

Let $v^{+*}_{\max}(s) = \sup_{\pi \in \Pi} v^{+\pi}_{\max}(s)$. We have for all states $s \in S$,

$$v^{+*}_{\max}(s) = \max_{a \in A_s} \max_{s' \in S} \left[r^{+}(s, a, s') + v^{+*}_{\max}(s') \right].$$

Under Condition 5.14 (Finite Total Positive Part Reward), the values $v^{+*}_{\max}(s)$ are finite since $v^{+*}_{\max}(s)$ is at most $(N-1)r_{\max}$ where N is the number of states.

Theorem 5.24. *Assume that Condition 4.1 (Nondecreasing Utility Function), Condition 2.1 (Finite Model), Condition 5.14 (Finite Total Positive Part Reward), and Condition 5.15 (Finite Total Negative Part Reward) hold. Then the optimal values $v_U^*(s)$ exist and are finite.*

Proof. The optimal values exist since according to Theorem 5.23, the values $v_U^\pi(s)$ exist for all policies under Condition 5.14.

To show that the optimal values are finite, first we notice that under Condition 5.14, for all policies $\pi \in \Pi$, all states $s \in S$, and all $T \in \mathbb{N}$, it holds that

$$v_{U,T}^\pi(s) = E^{s,\pi}[U(w_T)] \leq U(v_{\max,T}^{+\pi}(s)),$$

since $w_T \leq v_{\max,T}^{+\pi}(s)$. Since $v_U^\pi(s)$ exists according to Theorem 5.23, it holds that

$$v_U^\pi(s) = \lim_{T\to\infty} v_{U,T}^\pi(s) \leq \lim_{T\to\infty} U(v_{\max,T}^{+\pi}(s)) = U(v_{\max}^{+\pi}(s)) \leq U(v_{\max}^{+*}(s)).$$

Since π is arbitrary, it holds that

$$v_U^*(s) \leq U(v_{\max}^{+*}(s)) < \infty.$$

Next, suppose that Condition 5.15 (Finite Total Negative Part Reward) holds for policy π. Then it holds that

$$v_{U,T}^\pi(s) \geq U(v_{\min,T}^{-\pi}(s)).$$

Therefore,

$$v_U^*(s) \geq v_U^\pi(s) = \lim_{T\to\infty} v_{U,T}^\pi(s) \geq \lim_{T\to\infty} U(v_{\min,T}^{-\pi}(s)) = U(v_{\max}^{-\pi}(s)) > -\infty. \qquad \square$$

5.5 Summary

In this chapter, we studied conditions for the existence and finiteness of optimal values. We identified that linear and exponential utility functions can be viewed as “landmarks” to characterize the properties of other risk-sensitive utility functions. Moreover, we developed two classes of conditions on the utility functions and on the MDP models, and each class covers a full spectrum of conditions. We proved the existence of values for stationary policies under proper combinations of these conditions (at least one from each class). We also conjectured that these combinations of conditions also ensure the existence and finiteness of optimal values.

CHAPTER VI

CONCLUSION AND FUTURE WORK

6.1 Summary

In this thesis, I studied risk-sensitive planning objectives in decision-theoretical planning. Following the MEU principle, risk-sensitive planning objectives incorporate the risk attitudes of human decision makers into planning, and thus overcome some of the shortcomings of risk-neutral planning objectives, which are the focus of current decision-theoretic planning research. Instead of starting from scratch, ideas from decision-theoretic planning under the MER objective can be adapted to decision-theoretic planning under risk-sensitive planning objectives. Using Markov decision processes as the formal model, I established theoretical properties and developed efficient algorithms that make this possible. I demonstrated that optimal plans under a risk-sensitive objective are different from those under a risk-neutral objective, and that such plans can be constructed efficiently.

Chapter 3 studied risk-sensitive planning with exponential utility functions, which model constant risk attitudes. I showed that existing decision-theoretic planners can be transformed to take into account constant risk attitudes. The transformed algorithms bear visual resemblance to the original algorithms but special conditions are needed to ensure their validity. Moreover, different versions of the transformation are needed if the transition probabilities are implicitly given: the pseudo-probability transformation for temporally extended probabilities and the pseudo-discount factor transformation for probabilities given in a factored form. I showed that the main symbolic strategies in decision-theoretic planning for solving large-scale planning problems can still be used for risk-sensitive planning with constant risk attitudes, and the risk-sensitive versions of algorithms using such strategies can be obtained using the transformation and its variants to transform their risk-neutral counterparts.

Chapter 4 studied risk-sensitive planning with more general utility functions, which model variable risk attitudes. Using a state-augmentation approach, I showed that a functional interpretation of value functions and piecewise linear approximation methods can be used to solve planning problems efficiently, based on backward induction (for finite planning horizons) and value iteration (for infinite planning horizons and asymptotically constant, linear, or exponential utility functions). For one-switch utility functions, I also obtained an exact method similar to backward induction, based on results for planning with general, exponential, and linear utility functions.

Chapter 5 studied risk-sensitive planning with arbitrary rewards, while Chapter 3 and Chapter 4 considered risk-sensitive planning with negative and positive models. For problems with arbitrary rewards, the theoretical foundation is incomplete. In this chapter, I proposed different sets of conditions that form a spectrum and showed that under these conditions, the values of stationary policies exist and are finite. I also conjectured that under these conditions, the values of all policies and thus the optimal values exist and are finite.

6.2 Future Work: Short Term

This thesis developed theories and algorithms for risk-sensitive planning. The treatment is, however, far from complete. Here we list some possibilities for future research.

6.2.1 Theoretical Results

The convergence rates of the algorithms we presented in this thesis, especially those based on value iteration, are unknown. Although we know that these methods converge to the optimal values or an optimal policy, and even that methods for exponential utility functions converge at a geometric rate, it is not clear how close the results are to the optimal ones. Recent results for the MER objective (Bonet, 2002) suggest that the convergence rate can be estimated by examining how the values change over time.

For planning with arbitrary rewards, we need to investigate the conjectures for the existence and finiteness of the optimal values. The concepts of recurrent and transient states need to be extended for the induced random processes under a general HR-policy.

6.2.2 Solution Methods

In Chapter 3, we applied the transformation approach to methods using symbolic strategies such as heuristic search, temporal abstraction, and state abstraction. Another class of methods for large-scale problems have numerical strategies such as function approximators and direct policy search. It is worthwhile considering methods that use numerical strategies, since they can be more efficient for some types of problems than symbolic strategies. The main challenge is to show under what conditions such methods converge.

In Chapter 4, we considered value iteration using functional value functions. I anticipate that functional value functions can also be used in methods for large-scale problems with a symbolic strategy, therefore extending the applicability of the state-augmentation approach. The main difficulty is how to manage the complexity of functional value functions.

6.2.3 Extensions

Utility functions are elicited from human decision makers to be used by autonomous agents acting on their behalf. It is often the case that we can determine the form of utility functions, but only with imprecise parameters. In this case, a research topic is to study the sensitivity of optimal plans with respect to these imprecise parameters, and to develop methods for planning with imprecise utility functions.

Reinforcement learning is the problem for an agent to learn how to act by acting in the environment and receiving feedbacks. Using MDPs as the formal model, reinforcement learning methods can be viewed as online versions of planning methods, and do not assume that the dynamics of the agent's interaction with its environment is known. On the other hand, the current state is often only partially known in real-world planning problems. Such problems are captured by partially observable MDP (POMDP) models. Extensions for reinforcement learning and POMDPs under MEU objectives therefore constitute one more step towards solving real-world planning problems.

6.3 Future Work: Long Term

In the long term, I am interested in building autonomous agents that are able to act intelligently and provide valuable services to people in a complex environment involving uncertainty while taking into account the preference structures of their human users. Besides risk attitudes, there are other aspects that affect the preference structures of human users, such as the tradeoff among multiple types of rewards, and the requirement of achieving goals.

In real applications, the decision maker often trades off different resources, such as product quality and cost in business situations, or energy and time for robot navigation on Mars. Executing actions then results in reward tuples. In this case, preference structures can usually be captured by multi-attribute utility functions, and the planning objective is to maximize the expected multi-attribute utility.

Many planning problems require the agent to achieve some desired goal, which can be predefined goal states or temporally extended goals. Goals are very important in classical AI planning but not in decision-theoretic planning, since the latter uses rewards exclusively in its planning objectives. Although in some problems, the rewards can be assigned so that a plan maximizing the expected reward (or utility) also achieves the goal. But this is not the case in general. On the other hand, people often prefer a plan that guarantees goal achievement if only a small amount of the expected reward (or utility) is sacrificed. Therefore we need a more thorough study of planning tasks that require goal achievements.

All these aspects, and likely more, need to be taken into consideration when building usable planning systems including decision support systems for real-world applications such as e-commerce or planetary rovers. This thesis takes a first step towards building such planning systems by integrating ideas from artificial intelligence planning, operations research, and utility theory. Our results showed that it is promising that complex human preference structures, such as risk attitudes, can be incorporated into decision-theoretic planning without compromising too much of their efficiency.

REFERENCES

Ávila-Godoy, Guadalupe; Brau, Agustin; and Fernández-Gaucherand, Emmanuel, 1997. Controlled Markov chains with discounted risk-sensitive criteria: Applications to machine replacement. In *Proceedings of the 36th Conference on Decision and Control (CDC-97).*

Ávila-Godoy, Micaela Guadalupe, 1999. *Controlled Markov Chains with Exponential Risk-Sensitive Criteria: Modularity, Structured Policies and Applications*. Ph.D. thesis, Department of Mathematics, University of Arizona.

Bahar, R. Iris; Frohm, Erica A.; Gaona, Charles M.; Hachtel, Gary D.; Macii, Enrico; Pardo, Abelardo; and Somenzi, Fabio, 1993. Algebraic decision diagrams and their applications. In *Proceedings of the 1993 IEEE/ACM International Conference on Computer-Aided Design (ICCAD-93)*, pages 188–191.

Barberá, Salvador; Hammond, Peter J.; and Seidl, Christian, eds., 1998. *Handbook of Utility Theory, Volume 1: Principles*. Kluwer Academic Publishers.

Barto, Andrew G.; Bradtke, Steven J.; and Singh, Satinder P., 1995. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72:81–138.

Baxter, Jonathan and Bartlett, Peter L., 2001. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350.

Baxter, Jonathan; Bartlett, Peter L.; and Weaver, Lex, 2001. Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381.

Bell, David E., 1988. One-switch utility functions and a measure of risk. *Management Science*, 34(12):1416–1424.

Bell, David E. and Fishburn, Peter C., 2001. Strong one-switch utility. *Management Science*, 47(4):601–604.

Bell, David E.; Raiffa, Howard; and Tversky, Amos, eds., 1988. *Decision Making: Descriptive, Normative, and Prescriptive Interactions*. Cambridge University Press.

Bertsekas, Dimitri P., 2001. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, 2nd edition.

Bertsekas, Dimitri P. and Tsitsiklis, John N., 1991. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595.

Bertsekas, Dimitri P. and Tsitsiklis, John N., 1996. *Neuro-Dynamic Programming*. Athena Scientific.

Blythe, Jim, 1997. *Planning under Uncertainty in Dynamic Domains*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Blythe, Jim, 1999. Decision-theoretic planning. *AI Magazine*, 20(2):37–54.

Bonet, Blai, 2002. Stochastic shortest-path problems and real-time DP: New theoretical results. Available from the Web.

Bonet, Blai and Geffner, Héctor, 2001. Heuristic search planner: 2.0. *AI Magazine*, 22(3):77–80.

Bonet, Blai and Geffner, Héctor, 2002. Solving stochastic shortest-path problems with RTDP. Technical report, Department of Computer Science, University of California at Los Angeles.

Bonet, Blai and Geffner, Héctor, 2003a. Faster heuristic search algorithms for planning with uncertainty and full feedback. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 1233–1238.

Bonet, Blai and Geffner, Héctor, 2003b. Labeled RTDP: Improving the convergence of real-time dynamic programming. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS-03)*, pages 12–21.

Bouakiz, Mokrane, 1985. *Risk-Sensitivity in Stochastic Optimization with Applications*. Ph.D. thesis, School of Industrial and Systems Engineering, Georgia Institute of Technology.

Bouakiz, Mokrane and Kebir, Youcef, 1995. Target-level criterion in Markov decision processes. *Journal of Optimization Theory and Applications*, 86(1):1–15.

Boutilier, Craig, 1997. Correlated action effects in decision theoretic regression. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pages 30–37.

Boutilier, Craig; Dean, Thomas; and Hanks, Steve, 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94.

Boutilier, Craig and Dearden, Richard, 1996. Approximating value trees in structured dynamic programming. In *Proceedings of the Thirteenth International Conference on Machine Learning (ICML-96)*.

Boutilier, Craig; Dearden, Richard; and Goldszmidt, Moisés, 1995. Exploiting structure in policy construction. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*.

Boutilier, Craig; Dearden, Richard; and Goldszmidt, Moisés, 2000. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121:49–107.

Brau-Rojas, Agustin, 1999. *Controlled Markov Chains with Risk-Sensitive Average Cost Criterion*. Ph.D. thesis, Department of Mathematics, University of Arizona.

Cavazos-Cadena, Rolando and Fernández-Gaucherand, Emmanuel, 1999. Controlled Markov chains with risk-sensitive criteria: Average cost, optimality equations, and optimal solutions. *Mathematics Methods of Operations Research*, 49:299–324.

Cavazos-Cadena, Rolando and Montes-de-Oca, Raúl, 2000a. Nearly optimal policies in risk-sensitive positive dynamic programming on discrete spaces. *Mathematics Methods of Operations Research*, 52:133–167.

Cavazos-Cadena, Rolando and Montes-de-Oca, Raúl, 2000b. Optimal stationary policies in risk-sensitive dynamic programs with finite state space and nonnegative rewards. *Applicationes Mathematicae*, 27(2):167–185.

Chawla, Jay P., 2000. *Optimal Risk Sensitive Control of Semi-Markov Decision Processes*. Ph.D. thesis, Institute of Systems Research, University of Maryland at College Park.

Chung, Kun-Jen and Sobel, Matthew J., 1987. Discounted MDP's: Distribution functions and exponential utility maximization. *SIAM Journal of Control and Optimization*, 35(1):49–62.

Cohn, Paul R.; Greenberg, Michael L.; Hart, David M.; and Howe, Adele E., 1989. Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine*, 10(3):32–48.

Coraluppi, Stefano P. and Marcus, Steven I., 1996. Risk-sensitive control of Markov decision processes. In *Proceedings of the 30th Annual Conference on Information Sciences and Systems (CISS-96)*, pages 934–939.

Corman, Thomas H.; Leiserson, Charles E.; and Rivest, Ronald L., 1990. *Introduction to Algorithms*. The MIT Press.

Corner, James L. and Corner, Patricia D., 1995. Characteristics of decisions in decision analysis practice. *The Journal of Operational Research Society*, 46:304–314.

Dean, Thomas; Kaelbling, Leslie Pack; Kirman, Jak; and Nicholson, Ann, 1995. Planning under time constraints in stochastic domains. *Artificial Intelligence*, 76(1-2):35–74.

Dean, Thomas and Kanazawa, Keiji, 1989. A model for reasoning about persistence and causation. *Computational Intelligence*, 5:142–150.

Dean, Thomas and Lin, Shieu-Hong, 1995. Decomposition techniques for planning in stochastic domains. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*.

Dechter, Rina, 1996. Bucket elimination: A unifying framework for probabilistic inference. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 211–219.

Denardo, Eric V. and Rothblum, Uriel G., 1979. Optimal stopping, exponential utility, and linear programming. *Mathematical Programming*, 16:228–244.

di Masi, Giovanni B. and Stettner, Lukasz, 1999. Risk-sensitive control of discrete-time Markov processes with infinite horizon. *SIAM Journal of Control and Optimization*, 38(1):61–78.

Dietterich, Thomas G., 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303.

Dolgov, Dmitri and Durfee, Edmund, 2004. Approximate probabilistic constraints and risk-sensitive optimization criteria in Markov decision processes. In *Proceedings of the Eighth International Symposium on Artificial Intelligence and Mathematics*. Fort Lauderdale, FL.

Eagle, James Norfleet, II, 1975. *A Utility Criterion for the Markov Decision Process*. Ph.D. thesis, Department of Engineering-Economic Systems,Stanford University.

Fainberg [sic], Eugene A., 1982. Controlled Markov processed with arbitrary numerical criteria. *Theory of Probability and its Applications*, 27(3):486–503.

Farquhar, Peter H. and Nakamura, Yutaka, 1987. Constant exchange risk properties. *Operations Research*, 35(2):206–214.

Feinberg, Eugene A., 2002. Total reward criteria. In Eugene A. Feinberg and Adam Shwartz, eds., *Handbook of Markov Decision Processes: Methods and Applications*, chapter 6, pages 173–208. Kluwer.

Feinberg, Eugene A. and Shwartz, Adam, eds., 2002. *Handbook of Markov Decision Processes: Methods and Applications*. Kluwer.

Feng, Zhengzhu and Hansen, Eric A., 2001. Approximate planning for factored POMDPs. In *Proceedings of the Sixth European Conference on Planning (ECP-01)*.

Feng, Zhengzhu and Hansen, Eric A., 2002. Symbolic heuristic search for factored markov decision processes. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*.

Feng, Zhengzhu; Hansen, Eric A.; and Zilberstein, Shlomo, 2003. Symbolic generalization for on-line planning. In *Proceedings of the Eighteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 209–216.

Fern, Alan; Yoon, SungWook; and Givan, Robert, 2003. Approximate policy iteration with a policy language bias. In *Advances in Neural Information Processing Systems 16 (NIPS-03)*.

Fikes, Richard E. and Nilsson, Nils J., 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208.

Fishburn, Peter C., 1967. Bounded expected utility. *The Annals of Mathematical Statistics*, 38(4):1054–1060.

Fishburn, Peter C., 1970. *Utility Theory for Decision Making*. John Wiley & Sons.

Fishburn, Peter C., 1975. Unbounded expected utility. *The Annals of Statistics*, 3(4):884–896.

French, Simon, 1986. *Decision Theory: An Introduction to the Mathematics of Rationality*. Prentice Hall.

Friedman, Milton and Savage, Leonard J., 1948. The utility analysis of choices involving risk. *The Journal of Political Economy*, 56(4):279–304.

Givan, Robert; Dean, Thomas; and Greig, Matthew, 2003. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 147:163–223.

Givan, Robert; Leach, Sonia; and Dean, Thomas, 2000. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1-2):71–109.

Goodwin, Richard T.; Akkiraju, Rama; and Wu, Frederick, 2002. A decision-support system for quote-generation. In *Proceedings of the Fourteenth Conference on Innovative Applications of Artificial Intelligence (IAAI-02)*, pages 830–837.

Guestrin, Carlos; Koller, Daphne; and Parr, Ronald, 2001. Max-norm projections for factored MDPs. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 673–680.

Guestrin, Carlos; Koller, Daphne; Parr, Ronald; and Venkataraman, Shobha, 2003. Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, 19:399–468.

Haddawy, Peter and Hanks, Steve, 1992. Representations for decision-theoretic planning: Utility functions for deadline goals. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*, pages 71–82.

Hansen, Eric A., 1994. Cost-effective sensing during plan execution. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, pages 1029–1035.

Hansen, Eric A., 1997. Markov decision processes with observation costs. Technical Report 97-01, Department of Computer Science, University of Massachusetts at Amherst.

Hansen, Eric A. and Feng, Zhengzhu, 2000. Dynamic programming for POMDPs using a factored state representation. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-00)*, pages 130–139.

Hansen, Eric A.; Zhou, Rong; and Feng, Zhengzhu, 2002. Symbolic heuristic search using decision diagrams. In *Proceedings of the Fifth International Symposium on Abstraction, Reformulation and Approximation (SARA-02)*.

Hansen, Eric A. and Zilberstein, Shlomo, 1998. Heuristic search in cyclic AND/OR graphs. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 412–418.

Hansen, Eric A. and Zilberstein, Shlomo, 1999a. A heuristic search algorithm for Markov decision problems. In *Proceedings of Bar-Ilan Symposium on the Foundation of Artificial Intelligence*.

Hansen, Eric A. and Zilberstein, Shlomo, 1999b. Solving Markov decision problems using heuristic search. In *Proceedings of AAAI Spring Symposium on Search Techniques from Problem Solving under Uncertainty and Incomplete Information, March, Stanford, California*.

Hansen, Eric A. and Zilberstein, Shlomo, 2001. LAO*: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, 129:35–62.

Hart, Peter E.; Nilsson, Nils J.; and Raphael, Bertram, 1968. A formal basis for the heuristic determination of minimum cost paths in graphs. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Hartfiel, Darald J., 2002. *Nonhomogeneous Matrix Products*. World Scientific.

Harvey, Charles, 1995. Proportional discounting of future costs and benefits. *Mathematics of Operations Research*, 20(2):381–399.

Hauskrecht, Milos; Meuleau, Nicolas; Kaelbling, Leslie Pack; Dean, Thomas; and Boutilier, Craig, 1998. Hierarchical solution of Markov decision processes using macro-actions. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*.

Hernández-Hernández, Daniel and Marcus, Steven I., 1996. Risk-sensitive control of Markov processes in countable state space. *Systems and Control Letters*, 29:147–155.

Hernández-Hernández, Daniel and Marcus, Steven I., 1999. Existence of risk-sensitive optimal stationary policies for controlled Markov processes. *Applied Mathematics and Optimization*, 40(3):273–285.

Hill, Theodore P. and Pestien, Victor C., 1987. The existence of good Markov strategies for decision processes with general payoffs. *Stochastic Processes and their Applications*, 24:61–76.

Hinderer, Karl, 1970. *Foundations of Non-Stationary Dynamic Programming with Discrete Time Parameter*. Springer-Verlag.

Hoey, Jesse; St. Aubin, Robert; Hu, Alan J.; and Boutilier, Craig, 1999. SPUDD: Stochastic planning using decision diagrams. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 279–288.

Hoffman, Jörg and Nebel, Bernhard, 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302.

Howard, Ronald A. and Matheson, James E., 1972. Risk-sensitive markov decision processes. *Management Science*, 18(7):356–369.

Jaquette, Stratton C., 1973. Markov decision processes with a new optimality criterion: Discrete time. *The Annals of Statistics*, 1(3):496–505.

Jaquette, Stratton C., 1976. A utility criterion for Markov decision processes. *Management Science*, 23(1):43–49.

Jia, Zhongxiao, 1998. Generalized block Lanczos methods for large unsymmetric eigenproblems. *Numerische Mathematik*, 80:239–266.

Kadota, Yoshinobu; Kurano, Masami; and Yasuda, Masami, 1994. Discounted Markov decision processes with general utility functions. In *Proceedings of the Third Conference of the Association of Asian-Pacific Operational Research Societies (APORS) within IFORS, July 26–29, 1994, Fukuoka, Japan*, pages 330–337.

Kadota, Yoshinobu; Kurano, Masami; and Yasuda, Masami, 1998a. On the general utility of discounted Markov decision processes. *The International Transactions in Operational Research*, 5(1):27–34.

Kadota, Yoshinobu; Kurano, Masami; and Yasuda, Masami, 1998b. Stopped decision processes in conjuction with general utility. In *Proceedings of the 7th International Conference on Stochastic Programming*.

Kaelbling, Lelie Pack; Littman, Michael L.; and Moore, Andrew W., 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Kahneman, Daniel; Slovic, Paul; and Tversky, Amos, eds., 1982. *Judgement under Uncertainty: Heuristics and Biases*. Cambridge University Press.

Kearns, Michael; Mansour, Yishay; and Ng, Andrew Y., 2002. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning*, 49(2-3):193–208.

Keeney, Ralph L. and Raiffa, Howard, 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons.

Kemeny, John G. and Snell, James Laurie, 1960. *Finite Markov Chains*. D. Van Nostrand Company.

Kemeny, John G.; Snell, James Laurie; and Knapp, Anthony W., 1976. *Denumerable Markov Chains*. Springer-Verlag, 2nd edition.

Kennan, John, 1981. The existence of expected utility maximizing decisions when utility is unbounded. *Econometrica*, 49(1):215–218.

Kerr, Andrew L., 1999. Utility maximising stochastic dynamic programming: An overview. In *Proceedings of the 34th Annual Conference of the Operational Research Society of New Zealand (ORSNZ-99), December 10-11, Hamilton, New Zealand*.

Köbberling, Veronika, 2002. Preference foundations for difference representations. Technical report, METEOR, Maastricht University, The Netherlands.

Koenig, Sven, 1992. *Optimal Probabilistic and Decision-Theoretic Planning Using Markovian Decision Theory*. Master's thesis, Department of Electrical Engineering and Computer Science, University of California at Berkeley.

Koenig, Sven, 1997. *Goal-Directed Acting with Incomplete Information*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Koenig, Sven and Liu, Yaxin, 1999. Sensor planning with non-linear utility functions. In *Proceedings of the Fifth European Conference on Planning (ECP-99)*, pages 265–277.

Koenig, Sven and Liu, Yaxin, 2002. The interaction of representations and planning objectives for decision-theoretic planning tasks. *Journal of Experimental and Theoretical Artificial Intelligence*, 14(4):303–326.

Koenig, Sven and Simmons, Reid G., 1994a. How to make reactive planners risk-sensitive (without altering anything). In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems (AIPS-94)*, pages 293–298.

Koenig, Sven and Simmons, Reid G., 1994b. Risk-sensitive planning with probabilistic decision graphs. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR-94)*, pages 2301–2308.

Koller, Daphne and Parr, Ronald, 1999. Computing factored value functions for policies in structured MDPs. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 1332–1339.

Koller, Daphne and Parr, Ronald, 2000. Policy iteration for factored MDPs. In *Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-00)*, pages 326–334.

Korf, Richard E., 1987. Planning as search: A quantitative approach. *Artificial Intelligence*, 33(1):65–88.

Korf, Richard E., 1990. Real-time heuristic search. *Artificial Intelligence*, 42(3):189–212.

Kreps, David M., 1977a. Decision problems with expected utility criteria, I: Upper and lower convergent utility. *Mathematics of Operations Research*, 2(1):45–53.

Kreps, David M., 1977b. Decision problems with expected utility criteria, II: Stationarity. *Mathematics of Operations Research*, 2(3):266–274.

Kreps, David M., 1978. Decision problems with expected utility criteria, III: Upper and lower transience. *SIAM Journal of Control and Optimization*, 16(3):420–428.

Laibson, David I., 1994. *Hyperbolic Discounting and Consumption*. Ph.D. thesis, Department of Economics, Massachusetts Institute of Technology.

Liu, Yan-Qun; Teo, Kok-Lay; and Yang, Xiao-Qi, 1999. Approximation methods for nonconvex curves. *European Journal of Operational Research*, 117:125–135.

Marcus, Steven I.; Fernández-Gaucherand, Emmanual; Hernández-Hernandez, Daniel; Coraluppi, Stefano; and Fard, Pedram, 1997. Risk sensitive Markov decision processes. In Christopher I. Byrnes; Biswa Nath Datta; Clyde F. Martin; and David S. Gilliam, eds., *Systems and Control in the Twenty-First Century*, pages 263–279. Birkhäuser.

McCarthy, John and Hayes, Patrick J., 1969. Some philosophical problems from the standpoint of artificial intelligence. In Bernald Meltzer and Donald Michie, eds., *Machine Intelligence*, pages 463–502. Edinburgh University Press.

Müller, Alfred, 2000. Expected utility maximization of optimal stopping problems. *European Journal of Operational Research*, 12:101–114.

Murthy, Sesh; Akkiraju, Rama; Goodwin, Richard; Keskinocak, Pinar; Rachlin, John; Wu, Frederick; Kumaran, Santhosh; Yeh, James; Fuhrer, Robert; Aggarwal, Alok; Sturzenbecker, Martin; Jayaraman, Ranga; and Daigle, Bob, 1999. Cooperative multi-objective decision-support for the paper industry. *Interface*, 29(5):5–30.

Nau, Dana S.; Cao, Yue; Lotem, Amnon; and Muñoz-Avila, Héctor, 1999. SHOP: Simple hierarchical ordered planner. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 968–973.

Ng, Andrew Y. and Jordan, Michael I., 2000. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-00)*.

Nilsson, Nils J., 1980. *Principles of Artificial Intelligence*. Morgan Kaufman.

Ohtsubo, Yoshio and Toyonaga, Kenji, 2002. Optimal policy for minimizing risk models in Markov decision processes. *Journal of Mathematical Analysis and Applications*, 271:66–81.

Parr, Ronald, 1998. *Hierarchical Control and Learning for Markov Decision Processes*. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California at Berkeley.

Parr, Ronald and Russell, Stuart, 1998. Reinforcement learning with hierarchies of machines. In *Advances in Neural Information Processing Systems 11 (NIPS-98)*, pages 1043–1049.

Patek, Stephen D., 2001. On terminating Markov decision processes with a risk averse objective function. *Automatica*, 37(9):1379–1386.

Patrascu, Relu; Poupart, Pascal; Schuurmans, Dale; Boutilier, Craig; and Guestrin, Carlos, 2002. Greedy linear value function approximation for factored Markov decision processes. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*.

Pearl, Judea, 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan-Kaufmann.

Pednault, Edwin P.D., 1989. ADL: Exploring the middle ground between STRIPS and the situation calculus. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR-89)*, pages 324–332.

Pell, Barney; Bernard, Douglas; Chien, Steve; Gat, Erann; Muscettola, Nicola; Nayak, P. Pandurang; Wagner, Michael; and Williams, Brian, 1998. An autonomous spacecraft agent prototype. *Autonomous Robotics*, 5:1–27.

Porteus, Evan L., 1975. On the optimality of structured policies in countable stage decision processes. *Management Science*, 22(2):148–157.

Poupart, Pascal; Boutilier, Craig; Schuurmans, Dale; and Patrascu, Relu, 2002. Piecewise linear value function approximation for factored MDPs. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*.

Pratt, John W., 1964. Risk aversion in the small and in the large. *Econometrica*, 32(1-2):122–136.

Precup, Doina, 2000. *Temporal Abstraction in Reinforcement Learning*. Ph.D. thesis, Department of Computer Science, University of Massachusetts at Amherst.

Precup, Doina; Sutton, Richard S.; and Singh, Satinder, 1997. Planning with closed-loop macro actions. In *Working Notes of the AAAI Fall Symposium on Model-Directed Autonomous Systems*, pages 70–76. Cambridge, MA.

Puterman, Martin L., 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Rote, Günter, 1992. The convergence rate of the sandwich algorithm for approximating convex functions. *Computing*, 48:337–361.

Russell, Stuart J. and Novig, Peter, 2002. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition.

Schäl, Manfred, 1981. Utility functions and optimal policies in sequential decision problems. In Otto Moeschlin and Diethard Pallaschke, eds., *Game Theory and Mathematical Economics*, pages 357–365. North-Holland Publishing Company.

Seneta, Eugene, 1981. *Non-Negative Matrices and Markov Chains*. Springer-Verlag.

Simmons, Reid; Krotkov, Eric; Chrisman, Lonnie; Cozman, Fabio; Goodwin, Richard; Hebert, Martial; Katragadda, Lalitesh; Koenig, Sven; Krishnaswamy, Gita; Shinoda, Yoshikazu; Whittaker, William; and Klarer, Paul, 1995. Experience with rover navigation for lunar-like terrains. In *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-95)*, pages 441–446.

Simon, Herbert A., 1947. *Administrative Behavior*. MacMillan.

Somenzi, Fabio, 2004. CUDD: CU decision diagram package, Release 2.4.0. `http://vlsi.colorado.edu/~fabio/CUDD/cuddIntro.html`.

Sondik, Edward J., 1971. *The Optimal Control of Partially Observable Markov Processes*. Ph.D. thesis, Stanford University.

St. Aubin, Robert; Hoey, Jesse; and Boutilier, Craig, 2000. APRICODD: Approximate policy construction using decision diagrams. In *Advances in Neural Information Processing Systems 13 (NIPS-00)*.

Sutton, Richard S. and Barto, Andrew G., 1998. *Reinforcement Learning: An Introduction*. The MIT Press.

Sutton, Richard S.; McAllester, David; Singh, Satinder; and Mansour, Yishay, 1999a. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12 (NIPS-99)*, pages 1057–1063.

Sutton, Richard S.; Precup, Doina; and Singh, Satinder P., 1999b. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211.

Tate, Austin, 1977. Generating project networks. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*, pages 888–893.

van de Kuilen, Gijs and Wakker, Peter P., 2004. Learning in the Allais paradox. Working paper, CREED, Department of Economics, University of Amsterdam, The Netherlands.

von Neumann, John and Morgenstern, Oskar, 1944. *Theory of Games and Economic Behavior*. Princeton University Press.

White, Douglas John, 1987. Utility, probabilistic constraints, mean and variance of discounted rewards in Markov decision processes. *OR Spektrum*, 9:13–22.

White, Douglas John, 1993. Minimising a threshold probability in discounted Markov decision processes. *Journal of Mathematical Analysis and Applications*, 173(634-646):634–646.

Whittle, Peter, 1996. Why discount? The rationale of discounting in optimisation problems. In Chris C. Heyde; Yuri V. Prohorov; Ronald Pyke; and Svetlosar T. Rachev, eds., *Athens Conference on Applied Probability and Time Series, Volume 1: Applied Probability*, volume 114 of *Lecture Notes in Statistics*, pages 354–360. Springer-Verlag.

Wu, Congbin and Lin, Yuanlie, 1999. Minimizing risk models in Markov decision processes with policies depending on target values. *Journal of Mathematical Analysis and Applications*, 231:47–67.

Xie, Aiguo and Beerel, Peter A., 2000. Implicit enumeration of strongly connected components and an application to formal verification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19:1225–1230.

Yu, Stella X.; Lin, Yuanlie; and Yan, Pingfan, 1998. Optimization models for the first arrival target distribution function in discrete time. *Journal of Mathematical Analysis and Applications*, 225:193–223.

Zilberstein, Shlomo; Washington, Richard; Bernstein, Daniel S.; and Mouaddib, Abdel-Illah, 2002. Decision-theoretic control of planetary rovers. In Michael Beetz; Joachim Hertzberg; Malik Ghallab; and Martha E. Pollack, eds., *Advances in Plan-Based Control of Robotic Agents*, volume 2466 of *Lecture Notes in Computer Science*, pages 270–289. Springer.