An Exact Algorithm for Solving MDPs under Risk-Sensitive Planning Objectives with One-Switch Utility Functions*

Yaxin Liu
Research Division
Fair Isaac Corporation
yaxin.liu@gmail.com

Sven Koenig
Computer Science Department
University of Southern California
skoenig@usc.edu

ABSTRACT

One-switch utility functions are an important class of nonlinear utility functions that can model human beings whose decisions change with their wealth level. We study how to maximize the expected utility for Markov decision problems with given one-switch utility functions. We first utilize the fact that one-switch utility functions are weighted sums of linear and exponential utility functions to prove that there exists an optimal policy that is both stationary and deterministic as the wealth level approaches negative infinity. We then develop a solution method, the backward-induction method, that starts with this policy and augments it for higher and higher wealth levels. Our backward-induction method determines maximal expected utilities in finite time, different from the previous functional value iteration method, that typically determines only approximately maximal expected utilities.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Dynamic Programming*

General Terms

Algorithms

Keywords

Decision Making, Functional Value Iteration, Markov Decision Problem, One-Switch Utility Function, Planning, Utility Theory

1. INTRODUCTION

High-stake planning situations are planning situations with the possibility of high wins and losses. Traditional decision-theoretic planners typically maximize the expected reward (MER planning objective). In contrast, human beings are often risk-averse in high-stake planning situations and then maximize their expected utility for nonlinear utility functions (MEU planning objective), which explains why researchers have recently been very interested in attempting to maximize the expected utility for nonlinear utility functions or, equivalently, risk-sensitive utility functions [12, 14, 10]. We model decision-theoretic planning problems as finite goal-directed Markov decision problems (GDMDPs), which are Markov decision problems with strictly negative rewards (pure costs) and goal states, in which execution stops. Our planning objective is to maximize the expected utility for one-switch utility functions since these risk-sensitive utility functions can model human beings that are risk-averse but become risk-neutral in the limit as their wealth level increases [1]. In contrast, other decision-theoretic planners with the MEU planning objective often use exponential utility functions [6], which cannot model human beings whose risk attitudes change with their wealth level. The only previous decision-theoretic planner that can use one-switch utility functions is based on functional value iteration [9] and typically determines only approximately maximal expected utilities (similar to value iteration for the MER planning objective). We therefore introduce the backward-induction method, that exploits the structure of one-switch utility functions to determine maximal expected utilities and an optimal policy in finite time (similar to policy iteration for the MER planning objective). We apply it to a painted blocksworld problem and compare the results to those of the functional value iteration method.

2. UTILITY FUNCTIONS

Imagine that you are a contestant on the TV show "Who Wants to be a Millionaire" and reached the one million dollar question with only the 50-50 lifeline remaining. Since you do not know the answer, you use this lifeline to narrow down the possible answers to two alternatives and then have to make a decision. You can either leave with $500,000 for sure. Or you can guess the answer and then win $1,000,000 with 50% probability (if you are correct) and $32,000 with 50% probability (if you are wrong). The expected reward of leaving is $500,000, while the expected reward of guessing is $516,000. Thus, you would need to guess the answer in order to maximize the expected reward. However, many contestants choose to leave in this situation.

Utility theory, a major branch of decision theory, explains this risk-averse behavior as follows [16]: Every human being has a monotonically non-decreasing utility function that maps their wealth level to the resulting real-valued utility. A human being maximizes the expected utility of their future wealth level rather than their expected future wealth level itself. The utility function determines their risk attitude. Linear utility functions imply a risk-

*We thank Craig Tovey and Anton Kleywegt, two operations researchers, for lots of advice while the first author wrote his dissertation and their careful proofreading of the proofs. This research was partly supported by NSF awards to Sven Koenig under contracts IIS-9984827 and IIS-0098807 and an IBM fellowship to Yaxin Liu. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies or the U.S. government.

Cite as: An Exact Algorithm for Solving MDPs under Risk-Sensitive Planning Objectives with One-Switch Utility Functions, Yaxin Liu and Sven Koenig, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

Table 1: Utility-Theoretic Analysis of the Game-Show Problem

Utility Function	Leave	Guess			Utility Difference (Leave − Guess)	Optimal Decision
	Utility of \$500,000 (with prob. 1.0)	Utility of \$32,000 (with prob. 0.5)	Utility of \$1,000,000 (with prob. 0.5)	Expected Utility		
$U_\ell(w) = w$	$w_0 + 500,000$	$w_0 + 32,000$	$w_0 + 1,000,000$	$w_0 + 516,000$	$-16,000$	Guess
$U_{\rm e}(w) = -\gamma^w$	$-0.6065\gamma^{w_0}$	$-0.9685\gamma^{w_0}$	$-0.3679\gamma^{w_0}$	$-0.6682\gamma^{w_0}$	$0.0617\gamma^{w_0}$	Leave
$U_{\rm 1s}(w) =$ $w - D\gamma^w$	$w_0 + 500,000$ $-0.6065D\gamma^{w_0}$	$w_0 + 32,000$ $-0.9685D\gamma^{w_0}$	$w_0 + 1,000,000$ $-0.3679D\gamma^{w_0}$	$w_0 + 516,000$ $-0.6682D\gamma^{w_0}$	$-16,000$ $+0.0617D\gamma^{w_0}$	$w_0 > 1.35\times10^6$: Guess $w_0 < 1.35\times10^6$: Leave

w_0 = the initial wealth level. $\gamma = 0.999999$. $D = 10^6$.

neutral risk attitude. A human being is risk-neutral iff they make decisions that maximize their expected future wealth level. The calculations for the game-show problem with the linear utility function $U_\ell(w) = w$ (where w is the wealth level) are shown in the first row of Table 1. The optimal decision is to guess independent of the initial wealth level. Concave utility functions imply a risk-averse risk attitude. A human being is risk-averse iff they make decisions that do not maximize their expected future wealth level provided that the variance of their future wealth level is sufficiently reduced. Researchers often assume for mathematical convenience that risk-averse human beings have (concave) exponential utility functions, which are of the form $U_{\rm e}(w) = -\gamma^w$ for parameter $0 < \gamma < 1$ [4]. The calculations for the game-show problem with the exponential utility function $U_{\rm e}(w) = -0.999999^w$ are shown in the second row of Table 1. The optimal decision is to leave independent of the initial wealth level, which is consistent with the decision of most contestants in this situation. In general, different human beings can have different utility functions and thus different risk attitudes. Thus, they can make different decisions.

The decisions of human beings with linear or exponential utility functions are independent of their initial wealth level and thus do not change as their wealth level increases, which is why these utility functions are also known as zero-switch utility functions [1]. In reality, the decisions of human beings often change with their wealth level, which is why it can be unrealistic to use linear and exponential utility functions. For example, the rewards in the game-show problem are high compared to the wealth level of average people, which explains why they are expected to be risk-averse in game shows. However, the rewards are low compared to the wealth level of billionaires, which explains why they are expected to be risk-neutral. It is more realistic to assume that human beings are always risk-averse but become risk-neutral in the limit as their wealth level increases, that the utility increases monotonically with their wealth level (since they can always give money away), and that their decision between any two alternatives changes at most once as their wealth level increases. Human beings whose behavior satisfies these assumptions have special kinds of one-switch utility functions which are of the form

$$U_{\rm 1s}(w) = w - D\gamma^w$$

for parameters $D > 0$ and $0 < \gamma < 1$, as illustrated in Figure 1(a), where the dashed line indicates the linear utility function. The parameter D provides an adjustable tradeoff between risk-neutrality (linear term) and risk-aversion (exponential term). One-switch utility functions were proposed in [1] and have been studied extensively with many applications in the decision analysis community [1, 11, 5, 2]. The calculations for the game-show problem with the one-switch utility function $U_{\rm 1s}(w) = w - 10^6 \times 0.999999^w$ are shown in the third row of Table 1. The optimal decision now depends on the initial wealth level. For example, the expected utility of leaving is -1.065×10^5, while the expected utility of guessing is

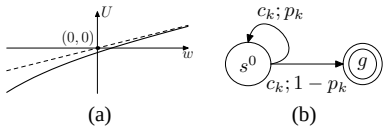


Figure 1: (a) One-Switch Utility Function (b) Two-State GDMDP

Action	Description	Success probability	Cost
1	do-it-yourself	0.25	\$ 100
2	hire a professional	0.95	\$ 1,000
3	buy a termite-free house ... and sell the infested one!	1.00	\$10,000

Figure 2: Termite Problem

-1.522×10^5 if your initial wealth level is zero. Thus, you would need to leave in order to maximize your expected utility. On the other hand, the exponential term of the one-switch utility function rapidly approaches zero as the initial wealth level increases and the linear term then dominates. Thus, you would eventually need to guess in order to maximize your expected utility. To determine at which initial wealth level w_0 you should switch from leaving to guessing, we solve the equation

$$-16,000 + 0.0617 \times 10^6 \cdot 0.999999^{w_0} = 0 \Rightarrow w_0 = 1.35 \times 10^6.$$

Thus, you should switch from leaving to guessing as your wealth level increases beyond about \$1,350,000.

3. GOAL-DIRECTED MDPS

The game-show problem involves only one action rather than a sequence of actions, the hallmark of artificial intelligence planning. Imagine therefore that you own a termite-infested wooden house but would like to own a termite-free house. The actions are given in Figure 2. The outcome of each action is either still owning a termite-infested house or a termite-free house. You can thus achieve the goal state only with a sequence of actions. For example, you could attempt to exterminate the termites yourself twice in a row and then buy a termite-free house, stopping after your first success. This policy costs you \$100 with probability 0.25, \$100 + \$100 = \$200 with probability $[1 - 0.25]\,0.25 = 0.19$, and \$100 + \$100 + \$10,000 = \$10,200 with probability $[1 - 0.25]\,[1 - 0.25] = 0.56$.

Sequential planning problems can be described with goal-directed Markov decision problems (GDMDPs). Formally, a GDMDP consists of a finite set of states S, a nonempty finite set of goal states $G \subseteq S$ and a finite set of actions A_s for each non-goal state $s \in S' = S \setminus G$. The agent starts execution at time step $t = 0$ and always chooses one action $a \in A_s$ to execute in its current state $s \in S$;. Its execution results with probability $P(s'|s, a)$ in a finite reward $r(s, a, s') < 0$ at the current time step and a transition to successor state $s' \in S$ at the next time step. The agent stops acting when it reaches a goal state and receives no more rewards

thereafter. We use s_t and a_t to denote the state and action at time step t, respectively. We use $r_t = r(s_t, a_t, s_{t+1})$ to denote the reward for executing action a_t. (We define $r_t = 0$ after the agent reaches a goal state.) Finally, we use $w_t = w_0 + \sum_{i=0}^{t-1} r_i$ to denote the wealth level at time step t directly before executing action a_t. The wealth level starts at the initial wealth level w_0 and then decreases over time since all rewards are negative. For example, Figure 1(b) shows how the termite problem can be modeled as a two-state GDMDP with three actions. One state is the start state s^0 (owning an infested house), and the other state is the goal state g (owning a termite-free house). In general, the agent has to choose among n actions to execute in state s^0, numbered from 1 to n. The arcs indicate the state transitions for each action $k = 1 \ldots n$ and are annotated with their probabilities p_k and rewards c_k. For the termite GDMDP, there are three actions. Their probabilities are their failure probabilities, and their rewards are their negative costs. For example, probability $p_1 = 0.75$ and reward $c_1 = -100$ for action 1, namely to exterminate the termites yourself.

Policies specify which actions an agent should execute. In the most general case, these action can probabilistically depend on the current state as well as all previous states and actions [15]. The agent should follow the policy that maximizes its expected utility. For all utility functions U and all policies π, we define the value $v_U^\pi(s, w) = \lim_{t\to\infty} E_{s,w}^\pi [U(w_t)]$ as the expected utility of an agent with initial state $s = s_0$ and initial wealth level $w = w_0$ that follows policy π. We also define the optimal value $v_U^*(s, w) = \max_\pi v_U^\pi(s, w)$ as the highest possible expected utility of an agent with initial state s and initial wealth level w. We assume that value $v_U^*(s, w)$ is finite for all states $s \in S$ and wealth levels w since it is otherwise impossible to compare policies [8]. We define an optimal policy π_U^* to be a policy with $v_U^{\pi_U^*}(s, w) = v_U^*(s, w)$ for all states $s \in S$ and wealth levels w. For the utility functions U_ℓ, U_e and U_1s, we refer to the MEU_ℓ (= MER), MEU_e and MEU_1s planning objectives, respectively, and replace the subscripts U in values $v_U^\pi(s, w)$ and $v_U^*(s, w)$ with ℓ, e and 1s, respectively. We use the shorthands $v_\ell^\pi(s) = v_\ell^\pi(s, 0)$ and $v_\ell^*(s) = v_\ell^*(s, 0) = \max_\pi v_\ell^\pi(s, 0) = \max_\pi v_\ell^\pi(s)$. Similarly, we use the shorthands $v_\mathrm{e}^\pi(s) = v_\mathrm{e}^\pi(s, 0)$ and $v_\mathrm{e}^*(s) = v_\mathrm{e}^*(s, 0) = \max_\pi v_\mathrm{e}^\pi(s, 0) = \max_\pi v_\mathrm{e}^\pi(s)$. We exploit the relationships between these values in the next section.

4. PLANNING OBJECTIVES

One-switch utility functions are linear combinations of linear and exponential utility functions:

$$U_\mathrm{1s}(w) = w - D\gamma^w = U_\ell(w) + D \cdot U_\mathrm{e}(w).$$

For all one-switch utility functions U_ℓ and all policies π, we thus have

$$\begin{aligned} v_\mathrm{1s}^\pi(s, w) &= \lim_{t\to\infty} E_{s,w}^\pi [U_\mathrm{1s}(w_t)] = \lim_{t\to\infty} E_{s,w}^\pi [U_\ell(w_t) + D \cdot U_\mathrm{e}(w_t)] \\ &= \lim_{t\to\infty} E_{s,w}^\pi [U_\ell(w_t)] + D \cdot \lim_{t\to\infty} E_{s,w}^\pi [U_\mathrm{e}(w_t)] \\ &= v_\ell^\pi(s, w) + D \cdot v_\mathrm{e}^\pi(s, w). \end{aligned} \quad (1)$$

We therefore need to consider the MEU_ℓ and MEU_e planning objectives. For these planning objectives, there always exists a stationary and deterministic (SD) policy that is optimal [3, 13]. An SD policy π maps every state $s \in S'$ to the policy $\pi(s) \in A_s$ that an agent in state s should execute independent of its wealth level. Thus, action $a_t = \pi(s_t)$ and reward r_t are independent of its initial wealth level. For the termite GDMDP, there are only three SD policies, namely $\pi_k(s^0) = k$ for actions $k = 1, 2, 3$.

First, we consider the MEU_ℓ planning objective: For the linear utility function U_ℓ and all SD policies π, we have

$$\begin{aligned} v_\ell^\pi(s, w) &= \lim_{t\to\infty} E_{s,w}^\pi [U_\ell(w_t)] = \lim_{t\to\infty} E_{s,w}^\pi [w_t] \\ &= \lim_{t\to\infty} E_{s,w}^\pi \left[w + \sum_{i=0}^{t-1} r_i \right] = w + \lim_{t\to\infty} E_s^\pi \left[\sum_{i=0}^{t-1} r_i \right] \\ &= w + v_\ell^\pi(s), \end{aligned} \quad (2)$$

where values $v_\ell^\pi(s)$ are independent of the wealth level w and satisfy the policy-evaluation equations [3]

$$\begin{aligned} v_\ell^\pi(s) &= 0 && \forall s \in G \\ v_\ell^\pi(s) &= \sum_{s' \in S} P(s'|s, \pi(s)) \left[r(s, \pi(s), s') + v_\ell^\pi(s') \right] && \forall s \in S'. \end{aligned} \quad (3)$$

Thus, $v_\ell^*(s, w) = w + v_\ell^*(s)$, where the optimal values $v_\ell^*(s)$ are also independent of the wealth level w and satisfy the optimality equations [3]

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

An agent in state $s \in S'$ with wealth level w follows an MEU_ℓ-optimal policy if it executes an action from $\arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) [r(s, a, s') + v_\ell^*(s')]$. For the termite GDMDP, the policy-evaluation equations are

$$v_\ell^{\pi_k}(g) = 0, \quad v_\ell^{\pi_k}(s^0) = p_k \left[c_k + v_\ell^{\pi_k}(s^0) \right] + [1 - p_k] \left[c_k + v_\ell^{\pi_k}(g) \right].$$

The values thus are

$$v_\ell^{\pi_k}(s^0) = \frac{c_k}{1 - p_k}.$$

The MEU_ℓ-optimal policy is π_1 since $v_\ell^{\pi_1}(s^0) = -400$, $v_\ell^{\pi_2}(s^0) = -1,052$ and $v_\ell^{\pi_3}(s^0) = -10,000$.

Second, we consider the MEU_e planning objective: For all exponential utility functions U_e and all SD policies π, we have similarly

$$\begin{aligned} v_\mathrm{e}^\pi(s, w) &= \lim_{t\to\infty} E_{s,w}^\pi [U_\mathrm{e}(w_t)] = \lim_{t\to\infty} E_{s,w}^\pi [-\gamma^{w_t}] \\ &= \lim_{t\to\infty} E_{s,w}^\pi \left[-\gamma^w \cdot \gamma^{\sum_{i=0}^{t-1} r_i} \right] = \gamma^w \cdot \lim_{t\to\infty} E_s^\pi \left[-\gamma^{\sum_{i=0}^{t-1} r_i} \right] \\ &= \gamma^w \cdot v_\mathrm{e}^\pi(s), \end{aligned} \quad (4)$$

where the values $v_\mathrm{e}^\pi(s)$ are independent of the wealth level w and satisfy the policy-evaluation equations [13]

$$\begin{aligned} v_\mathrm{e}^\pi(s) &= -1 && \forall s \in G \\ v_\mathrm{e}^\pi(s) &= \sum_{s' \in S} P(s'|s, \pi(s)) \gamma^{r(s, \pi(s), s')} \cdot v_\mathrm{e}^\pi(s') && \forall s \in S', \end{aligned} \quad (5)$$

provided that the values $v_\mathrm{e}^\pi(s)$ are finite for all states $s \in S'$. Thus, $v_\mathrm{e}^*(s, w) = \gamma^w \cdot v_\mathrm{e}^*(s)$, where the optimal values $v_\mathrm{e}^*(s)$ are also independent of the wealth level w and satisfy the optimality equations [13]

$$\begin{aligned} v_\mathrm{e}^*(s) &= -1 && \forall s \in G \\ v_\mathrm{e}^*(s) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \gamma^{r(s, a, s')} \cdot v_\mathrm{e}^*(s') && \forall s \in S', \end{aligned}$$

provided that the optimal values $v_\mathrm{e}^*(s)$ are finite for all states $s \in S'$. An agent in state $s \in S'$ with wealth level w follows an MEU_e-optimal policy if it executes an action from $\arg\max_{a \in A_s} \sum_{s' \in S} P(s'|s, a) \gamma^{r(s, a, s')} \cdot v_\mathrm{e}^*(s')$. For the termite GDMDP, the policy-evaluation equations are

$$v_\mathrm{e}^{\pi_k}(g) = -1 \text{ and } v_\mathrm{e}^{\pi_k}(s^0) = p_k \gamma^{c_k} v_\mathrm{e}^{\pi_k}(s^0) + [1 - p_k] \gamma^{c_k} v_\mathrm{e}^{\pi_k}(g),$$

provided that the values $v_\mathrm{e}^{\pi_k}(s^0)$ are finite, which is the case if

$p_k\gamma^{c_k} < 1$. (Otherwise, $v_{\mathrm{e}}^{\pi_k}(s^0) = -\infty$.) The values thus are

$$v_{\mathrm{e}}^{\pi_k}(s^0) = \begin{cases} -\frac{[1-p_k]\gamma^{c_k}}{1-p_k\gamma^{c_k}} & p_k\gamma^{c_k} < 1 \\ -\infty & p_k\gamma^{c_k} \geq 1. \end{cases}$$

Assume that your exponential utility function is $U_{\mathrm{e}}(w) = -0.997^w$. Then, the $\mathsf{MEU}_{\mathrm{e}}$-optimal policy is π_3 since $v_{\mathrm{e}}^{\pi_1}(s^0) = v_{\mathrm{e}}^{\pi_2}(s^0) = -\infty$ and $v_{\mathrm{e}}^{\pi_3}(s^0) = -1.1179 \times 10^{13}$.

Third, we consider the $\mathsf{MEU}_{1\mathrm{s}}$ planning objective: For all one-switch utility functions $U_{1\mathrm{s}}$ and all SD policies π, we can decompose the values $v_{1\mathrm{s}}^{\pi}(s,w)$ according to Eq. (1), Eq. (2) and Eq. (4) as follows:

$$v_{1\mathrm{s}}^{\pi}(s,w) = w + v_{\ell}^{\pi}(s) + D\gamma^w \cdot v_{\mathrm{e}}^{\pi}(s). \tag{6}$$

For the termite GDMDP, these equations are

$$v_{1\mathrm{s}}^{\pi_k}(g,w) = U_{1\mathrm{s}}(w) = w - D\gamma^w$$

$$v_{1\mathrm{s}}^{\pi_k}(s^0,w) = \begin{cases} w + \frac{c_k}{1-p_k} - D\gamma^w \cdot \frac{[1-p_k]\gamma^{c_k}}{1-p_k\gamma^{c_k}} & p_k\gamma^{c_k} < 1 \\ -\infty & p_k\gamma^{c_k} \geq 1. \end{cases}$$

Assume that your one-switch utility function is $U_{1\mathrm{s}}(w) = w - 10^{-9} \times 0.997^w$ and your initial wealth level is $w = 0$ (due to your mortgage debt). Then, the $\mathsf{MEU}_{1\mathrm{s}}$-optimal SD policy is π_3 since $v_{1\mathrm{s}}^{\pi_1}(s^0,0) = v_{1\mathrm{s}}^{\pi_2}(s^0,0) = -\infty$ and $v_{1\mathrm{s}}^{\pi_3}(s^0,0) = -21,179$.

For the $\mathsf{MEU}_{1\mathrm{s}}$ planning objective, there does not necessarily exist an SD policy that is optimal but there always exists an augmented stationary and deterministic (ASD) policy that is optimal [9]. An ASD policy maps every combination of a state $s \in S'$ and wealth level w to the action $\pi(s,w) \in A_s$ that an agent in state s with wealth level w should execute. Thus, action $a_t = \pi(s_t,w_t) = \pi(s_t, w_0 + \sum_{i=0}^{t-1} r_i)$ and reward r_t are no longer independent of its initial wealth level w_0. For the termite GDMDP, assume again that your one-switch utility function is $U_{1\mathrm{s}}(w) = w - 10^{-9} \times 0.997^w$ and your initial wealth level is zero. The policy of attempting to exterminate the termites yourself twice in a row and then buy a termite-free house, stopping after your first success, can be formulated as an ASD policy π with actions $\pi(s^0,0) = \pi(s^0,-100) = 1$ and $\pi(s^0,-200) = 3$. Its value is $v_{1\mathrm{s}}^{\pi}(s^0) = 0.25 \cdot U_{1\mathrm{s}}(-100) + 0.19 \cdot U_{1\mathrm{s}}(-200) + 0.56 \cdot U_{1\mathrm{s}}(-10,200) = -17,193$. Thus, this ASD policy is better than the $\mathsf{MEU}_{1\mathrm{s}}$-optimal SD policy π_3.

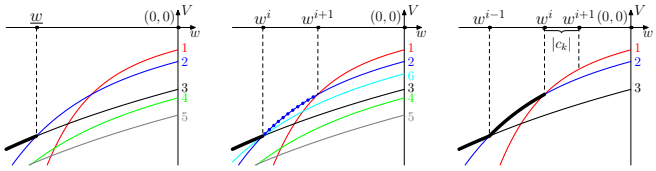
We now generalize Eq. (6) to ASD policies by reformulating special cases of Theorems 1 and 2 from [9], which hopefully make them more accessible to the reader. For all ASD policies π and all wealth levels w, we define the policy π_w that satisfies $\pi_w(s,w') = \pi(s,w+w')$ for all wealth levels w'. (If π is an SD policy, then $\pi_w = \pi$.) The example ASD policy for the termite GDMDP has actions $\pi_{-100}(s^0,0) = \pi(s^0,-100) = 1$ and $\pi_{-100}(s^0,-100) = \pi(s^0,-200) = 3$.

An agent with initial wealth level w that follows policy π executes action $a_t = \pi(s_t, w + \sum_{i=0}^{t-1} r_i)$ at time step t, while an agent with initial wealth level 0 that follows policy π_w executes action $a_t = \pi_w(s_t, \sum_{i=0}^{t-1} r_i) = \pi(s_t, w + \sum_{i=0}^{t-1} r_i)$ at time step t. These actions are the same and thus the probability distributions over the successor states and rewards are also the same. Thus, we have

$$\begin{aligned} v_{\ell}^{\pi}(s,w) &= \lim_{t\to\infty} E_{s,w}^{\pi}\left[w + \sum_{i=0}^{t-1} r_i\right] = w + \lim_{t\to\infty} E_{s}^{\pi_w}\left[\sum_{i=0}^{t-1} r_i\right] \\ &= w + v_{\ell}^{\pi_w}(s,0) = w + v_{\ell}^{\pi_w}(s) \end{aligned} \tag{7}$$

and

$$v_{\mathrm{e}}^{\pi}(s,w) = \lim_{t\to\infty} E_{s,w}^{\pi}\left[-\gamma^w \cdot \gamma^{\sum_{i=0}^{t-1} r_i}\right] = \gamma^w \cdot \lim_{t\to\infty} E_{s}^{\pi_w}\left[-\gamma^{\sum_{i=0}^{t-1} r_i}\right]$$



(a) Determine $\underline{w}$ (b) Determine w^{i+1} (c) Maintain Property 2

Figure 3: Illustration of Backward Induction

$$= \gamma^w \cdot v_{\mathrm{e}}^{\pi_w}(s,0) = \gamma^w \cdot v_{\mathrm{e}}^{\pi_w}(s). \tag{8}$$

For all one-switch utility functions $U_{1\mathrm{s}}$ and all ASD policies π, we can therefore decompose the value $v_{1\mathrm{s}}^{\pi}(s,w)$ according to Eq. (1), Eq. (7) and Eq. (8) as follows:

$$v_{1\mathrm{s}}^{\pi}(s,w) = w + v_{\ell}^{\pi_w}(s) + D\gamma^w \cdot v_{\mathrm{e}}^{\pi_w}(s). \tag{9}$$

The values $v_{1\mathrm{s}}^{\pi}(s,w)$ satisfy policy-evaluation equations that are implied by [9] but were not explicitly stated there

$$\begin{aligned} v_{1\mathrm{s}}^{\pi}(s,w) &= U_{1\mathrm{s}}(w) = w - D\gamma^w && \forall s \in G, \forall w \\ v_{1\mathrm{s}}^{\pi}(s,w) &= \sum_{s'\in S} P\big(s'|s,\pi(s,w)\big) v_{1\mathrm{s}}^{\pi}\big(s', w + r\big(s,\pi(s,w),s'\big)\big) \\ & && \forall s \in S', \forall w, \end{aligned}$$

provided that the values $v_{1\mathrm{s}}^{\pi}(s,w)$ are finite for all states $s \in S'$ and all wealth levels w. The optimal values $v_{1\mathrm{s}}^{*}(s,w)$ satisfy optimality equations that are again implied by [9]

$$\begin{aligned} v_{1\mathrm{s}}^{*}(s,w) &= U_{1\mathrm{s}}(w) = w - D\gamma^w && \forall s \in G, \forall w \\ v_{1\mathrm{s}}^{*}(s,w) &= \max_{a\in A_s} \sum_{s'\in S} P(s'|s,a) v_{1\mathrm{s}}^{*}\big(s', w + r(s,a,s')\big) \\ & && \forall s \in S', \forall w, \end{aligned} \tag{10}$$

provided that the optimal values $v_{1\mathrm{s}}^{*}(s,w)$ are finite for all state $s \in S'$ and all wealth levels w. Then, an agent in state $s \in S'$ with wealth level w follows an $\mathsf{MEU}_{1\mathrm{s}}$-optimal policy if it executes an action from $\arg\max_{a\in A_s} \sum_{s'\in S} P(s'|s,a) v_{1\mathrm{s}}^{*}(s', w + r(s,a,s'))$. Therefore, all that is left to do is to determine the optimal values $v_{1\mathrm{s}}^{*}(s,w)$. The only previous approach for this purpose that we know of is based on functional value iteration [9] and typically determines the optimal values $v_{1\mathrm{s}}^{*}(s,w)$ only in the limit although it needs to terminate in finite time in practice. However, the approach does not provide a termination condition for improving the values nor an error bound on the resulting policies. We therefore introduce that backward-induction method that exploits the relationship between one-switch utility functions and exponential utility functions to determine the optimal values $v_{1\mathrm{s}}^{*}(s,w)$ and thus an $\mathsf{MEU}_{1\mathrm{s}}$-optimal policy in finite time.

5. INDUCTIVE FOUNDATION

We now establish the induction foundation for the backward-induction method. We utilize the fact that one-switch utility functions are weighted sums of linear and exponential utility functions to prove that there exists an SD policy that is an $\mathsf{MEU}_{1\mathrm{s}}$-optimal policy as the wealth level approaches negative infinity. The backward-induction method then starts with this policy and augments it for higher and higher wealth levels. We illustrate the backward-induction method in Figure 3 for a general two-state GDMDP with initial wealth level zero, where the graphs are the value functions $v_{1\mathrm{s}}^{\pi_k}(s^0,\cdot)$ of the policies π_k. The following lemma relates the optimal values $v_{1\mathrm{s}}^{*}(s,w)$ and $v_{\mathrm{e}}^{*}(s)$ and uses the fact

that we have for all MEU_{1s}-optimal ASD policies π^*_{1s} according to Eq. (9)

$$v^*_{1s}(s,w) = v^{\pi^*_{1s}}_{1s}(s,w) = w + v^{(\pi^*_{1s})_w}_{\ell}(s) + D\gamma^w v^{(\pi^*_{1s})_w}_{e}(s). \quad (11)$$

LEMMA 1. $\lim_{w\to-\infty} v^*_{1s}(s,w)\gamma^{-w} = Dv^*_e(s)$ *for all states* $s \in S$.

PROOF. For all MEU_{1s}-optimal ASD policies π^*_{1s}, we have according to Eq. (11) and the fact that $v^\pi_\ell(s) \le v^*_\ell(s)$ and $v^\pi_e(s) \le v^*_e(s)$ for all policies π

$$\begin{aligned} v^*_{1s}(s,w)\gamma^{-w} &= v^{\pi^*_{1s}}_{1s}(s,w)\gamma^{-w} = w\gamma^{-w} + v^{(\pi^*_{1s})_w}_{\ell}(s)\gamma^{-w} + Dv^{(\pi^*_{1s})_w}_{e}(s) \\ &\le w\gamma^{-w} + v^*_\ell(s)\gamma^{-w} + Dv^*_e(s). \end{aligned}$$

and thus

$$\limsup_{w\to-\infty} v^*_{1s}(s,w)\gamma^{-w} \le \lim_{w\to-\infty} \left[w\gamma^{-w} + v^*_\ell(s)\gamma^{-w} + Dv^*_e(s)\right] = Dv^*_e(s).$$

On the other hand, for all MEU_e-optimal SD policies π^*_e, we have according to Eq. (9) and the fact that $v^*_{1s}(s,w) \ge v^\pi_{1s}(s,w)$ for all policies π

$$\begin{aligned} v^*_{1s}(s,w)\gamma^{-w} &\ge v^{\pi^*_e}_{1s}(s,w)\gamma^{-w} = w\gamma^{-w} + v^{\pi^*_e}_{\ell}(s)\gamma^{-w} + Dv^{\pi^*_e}_{e}(s) \\ &= w\gamma^{-w} + v^{\pi^*_e}_{\ell}(s)\gamma^{-w} + Dv^*_e(s) \end{aligned}$$

and thus

$$\liminf_{w\to-\infty} v^*_{1s}(s,w)\gamma^{-w} \ge \lim_{w\to-\infty} \left[w\gamma^{-w} + v^{\pi^*_e}_{\ell}(s)\gamma^{-w} + Dv^*_e(s)\right] = Dv^*_e(s).$$

Therefore, the lemma holds. □

The lemma implies that an MEU_{1s}-optimal policy is also MEU_e-optimal in the limit, which is not surprising since the exponential term in Eq. (11) grows faster than the linear term. However, not every MEU_e-optimal policy is also MEU_{1s}-optimal in the limit. For a general two-state GDMDP, assume that there are only two actions and both of the corresponding SD policies π_1 and π_2 are MEU_e-optimal. Thus, we have for actions $k = 1, 2$

$$v^*_e(s^0) = v^{\pi_k}_e(s^0) = -\frac{[1-p_k]\gamma^{c_k}}{1-p_k\gamma^{c_k}} < -1,$$

provided that $p_k\gamma^{c_k} < 1$, which implies

$$p_k = \frac{1 + v^*_e(s^0)\gamma^{-c_k}}{1 + v^*_e(s^0)} \quad \text{and} \quad c_k \in \left[\log_\gamma\left(-v^*_e(s^0)\right), 0\right).$$

Thus, there are combinations of probabilities p_k and rewards c_k that achieve the same optimal value $v^*_e(s^0)$ but differ in their values $v^{\pi_k}_\ell(s^0) = \frac{c_k}{1-p_k}$. Only the policy π_k with the highest value $v^{\pi_k}_\ell(s^0)$ can possibly be MEU_{1s}-optimal according to Eq. (11). The following lemma formalizes this observation. It defines an auxiliary GDMDP such that all policies for the auxiliary GDMDP are MEU_e-optimal for the given GDMDP. The policy that is MEU_ℓ-optimal for the auxiliary GDMDP then is MEU_{1s}-optimal for the given GDMDP for all wealth levels no higher than some wealth level threshold.

LEMMA 2. *For all* MEU_{1s}*-optimal policies* π^*_{1s}*, there exists a wealth level threshold* $\underline{w}$ *such that it holds for all wealth levels* $w \le \underline{w}$ *that*

1. $v^{(\pi^*_{1s})_w}_e(s) = v^*_e(s)$ *for all states* $s \in S'$.
2. $v^{(\pi^*_{1s})_w}_\ell(s) = v^{\pi^{**}_e}_\ell(s)$ *for all states* $s \in S'$*, where* π^{**}_e *is any SD* MEU_ℓ*-optimal policy for the auxiliary GDMDP which is the same as the original GDMDP except that the agent chooses its actions only from the* MEU_e*-optimal actions, that is, the sets*

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

3. $v^*_{1s}(s,w) = v^{\pi^{**}_e}_{1s}(s,w)$ *for all states* $s \in S'$*, meaning that* π^{**}_e *is an* MEU_{1s}*-optimal policy for all wealth levels* $w \le \underline{w}$.

The proof of the lemma can be found in [7]. We need to determine the wealth level threshold $\underline{w}$ to establish the induction foundation, as shown in Figure 3(a). For a general two-state GDMDP, assume that there are only two actions and that π_1 is an MEU_{1s}-optimal policy for all wealth levels $w \le \underline{w}$, whereas π_2 is not. Assume further that π_1 is no longer an MEU_{1s}-optimal policy for wealth levels that are higher than the wealth level threshold $\underline{w}$ by a positive infinitesimal. Then, we have for all wealth levels $w \le \underline{w}$ according to Eq. (11) and the fact that $(\pi_1)_w = \pi_1$ (since π_1 is an SD policy)

$$v^*_{1s}(s^0,w) = v^{\pi_1}_{1s}(s^0,w) = w + v^{\pi_1}_\ell(s^0) + D\gamma^w v^{\pi_1}_e(s^0). \quad (12)$$

Now consider any wealth level w that is higher than the wealth level threshold $\underline{w}$ by a positive infinitesimal but no higher than $\underline{w} - c_k$ for all actions $k = 1, 2$. Then, $v^*_{1s}(s^0, w + c_k) = v^{\pi_1}_{1s}(s^0, w + c_k)$ (since $w + c_k \le \underline{w}$) and we can rewrite Eq. (10) as follows:

$$\begin{aligned} v^*_{1s}(s^1,w) &= w - D\gamma^w \\ v^*_{1s}(s^0,w) &= \max_{k=1,2} \left[p_k v^*_{1s}(s^0, w+c_k) + [1-p_k]\, v^*_{1s}(s^1, w+c_k)\right] \\ &= \max_{k=1,2} \left[p_k v^{\pi_1}_{1s}(s^0, w+c_k) + [1-p_k]\left[w + c_k - D\gamma^{w+c_k}\right]\right] \\ &= \max_{k=1,2} \left[w + q^{\pi_1}_\ell(s^0,k) + D\gamma^w q^{\pi_1}_e(s^0,k)\right], \end{aligned}$$

where the last step uses both Eq. (12) and the definitions of the values $q^{\pi_1}_\ell(s^0,k) = c_k + p_k v^{\pi_1}_\ell(s^0)$ and $q^{\pi_1}_e(s^0,k) = \gamma^{c_k}\left[p_k - 1 + p_k v^{\pi_1}_e(s^0)\right]$ for $k = 1, 2$. It holds that $q^{\pi_1}_\ell(s^0,1) = v^{\pi_1}_\ell(s^0)$ according to Eq. (3) and $q^{\pi_1}_e(s^0,1) = v^{\pi_1}_e(s^0)$ according to Eq. (5). π_2 is an MEU_{1s}-optimal policy for wealth level w according to our assumptions, whereas π_1 is not. Thus, we have

$$\begin{aligned} w + q^{\pi_1}_\ell(s^0,2) + D\gamma^w q^{\pi_1}_e(s^0,2) &> w + q^{\pi_1}_\ell(s^0,1) + D\gamma^w q^{\pi_1}_e(s^0,1) \\ &= w + v^{\pi_1}_\ell(s^0) + D\gamma^w v^{\pi_1}_e(s^0) \end{aligned}$$

or, equivalently,

$$\begin{aligned} q^{\pi_1}_\ell(s^0,2) - v^{\pi_1}_\ell(s^0) &> D\gamma^w \left[v^{\pi_1}_e(s^0) - q^{\pi_1}_e(s^0,2)\right] \\ w &> \log_\gamma\left(\frac{1}{D}\cdot\frac{q^{\pi_1}_\ell(s^0,2) - v^{\pi_1}_\ell(s^0)}{v^{\pi_1}_e(s^0) - q^{\pi_1}_e(s^0,2)}\right). \end{aligned}$$

The wealth level threshold $\underline{w}$ can thus be set to the right-hand side of the last inequality. We actually set it to the minimum of the right-hand side of the last inequality and the initial wealth level since the agent only encounters wealth levels no higher than its initial wealth level. If the argument of the logarithm is non-positive and the logarithm thus is undefined, then policy π_1 is MEU_{1s}-optimal for all wealth levels and we set the wealth level threshold to the initial wealth level. The general case is just slightly more complex than for the two-state GDMDP since one needs to minimize over all states $s \in S'$ and all actions $a \in A_s \setminus A^*_e(s)$. The following theorem summarizes the discussion. It uses the following definitions for all states $s \in S'$ and actions $a \in A_s$:

$$\begin{aligned} q^*_e(s,a) &= \sum_{s'\in S} P(s'|s,a)\gamma^{r(s,a,s')} v^*_e(s') \\ q^{\pi^{**}_e}_\ell(s,a) &= \sum_{s'\in S} P(s'|s,a)\left[r(s,a,s') + v^{\pi^{**}_e}_\ell(s')\right]. \end{aligned}$$

THEOREM 3. *The statements in Lemma 2 hold for*

$$\underline{w} = \min_{s \in S'} \min_{a \in A_s \setminus A^*_{\mathsf{e}}(s)} \log_\gamma \left(\max \left[\gamma^{w_0}, \frac{1}{D} \cdot \frac{q_\ell^{\pi_{\mathsf{e}}^{**}}(s,a) - v_\ell^{\pi_{\mathsf{e}}^{**}}(s)}{v_{\mathsf{e}}^*(s) - q_{\mathsf{e}}^*(s,a)} \right] \right).$$

The proof of the theorem can be found in [7]. The maximum in the definition of the wealth level threshold $\underline{w}$ takes care of the case where the numerator is negative or the wealth level threshold would otherwise be higher than the initial wealth level w_0. One needs to determine $A^*_{\mathsf{e}}(s)$, $v^*_{\mathsf{e}}(s)$, π^{**}_{e} and $v_\ell^{\pi^{**}_{\mathsf{e}}}(s)$ to calculate the wealth level threshold. The action sets $A^*_{\mathsf{e}}(s)$ and the optimal values $v^*_{\mathsf{e}}(s)$ can be calculated with the $\mathsf{MEU}_{\mathsf{e}}$ version of policy iteration [13], and then the policy π^{**}_{e} and the optimal values $v_\ell^{\pi^{**}_{\mathsf{e}}}(s)$ can be calculated with the MEU_ℓ version of policy iteration [3]. One should not use versions of value iteration for this purpose since they typically determine optimal values only approximately. For the termite GDMDP, we have shown that π_3 is the only SD $\mathsf{MEU}_{\mathsf{e}}$-optimal policy and thus $\pi^{**}_{\mathsf{e}} = \pi_3$ and $A^*_{\mathsf{e}}(s^0) = \{3\}$. Then, we have $v_\ell^{\pi^{**}_{\mathsf{e}}}(s^0) = -10,000$, $q_\ell^{\pi^{**}_{\mathsf{e}}}(s^0,1) = -7,600$, $q_\ell^{\pi^{**}_{\mathsf{e}}}(s^0,2) = -1,500$, $v^*_{\mathsf{e}}(s^0) = -1.1179 \times 10^{13}$, $q^*_{\mathsf{e}}(s^0,1) = -1.1323 \times 10^{13}$ and $q^*_{\mathsf{e}}(s^0,2) = -1.1278 \times 10^{13}$. Therefore, the wealth level threshold is $\underline{w} = -1482.0$.

6. BACKWARD-INDUCTION METHOD

Algorithm 1 (BackwardInductionOneSwitch) shows our backward-induction method, which is based on the inductive foundation from the previous section and Eq. (10). There are two main differences to the common backward-induction method for solving GDMDPs with finite planning horizons [15]. First, the inductive foundation of the backward-induction method for solving GDMDPs with finite planning horizons is trivially provided by values that are all zero, while the inductive foundation of our backward-induction method is provided by Theorem 3. Thus, our backward-induction method first determines the wealth level threshold $\underline{w}$ and, at the same time, the optimal values $v^*_{1\mathsf{s}}(s,w)$ for all states $s \in S'$ and all wealth levels w with $w \le \underline{w}$. Second, the backward-induction method for solving GDMDPs with finite planning horizons starts at the planning horizon $t = T$, then decreases the time step t with a fixed step size of one, and ends at time step 0, while our backward-induction method starts at the wealth level threshold $w = \underline{w}$, then increases the wealth level w with a variable step size (that is controlled by a priority queue with the wealth level w as the key), and ends at the initial wealth level w_0.

We now explain the backward-induction method and show, at the same time, that the $\mathsf{MEU}_{1\mathsf{s}}$-optimal value functions $v^*_{1\mathsf{s}}(s,\cdot)$ are piecewise one-switch functions with a finite number of segments. The backward-induction method represents the $\mathsf{MEU}_{1\mathsf{s}}$-optimal value functions (one for each state) and an $\mathsf{MEU}_{1\mathsf{s}}$-optimal policy as a finite list of tuples $(w^i(s), v^i_\ell(s), v^i_{\mathsf{e}}(s), a^i(s))$, which represent that $v^*_{1\mathsf{s}}(s,w) = w + v^i_\ell(s) + D\gamma^w v^i_{\mathsf{e}}(s)$ and that $\pi^*_{1\mathsf{s}}(s,w) = a^i(s)$ for all wealth levels w with $w \in \big(w^i(s), w^{i+1}(s)\big]$. In this case, we say that $a^i(s)$ is an $\mathsf{MEU}_{1\mathsf{s}}$-optimal action for wealth level w. *VAList* is a data structure that contains all of these tuples after the termination of the backward-induction method. GetValues(*VAList*, s, w) retrieves the values $v^i_\ell(s)$ and $v^i_{\mathsf{e}}(s)$ from the tuple $\big(w^i(s), v^i_\ell(s), v^i_{\mathsf{e}}(s), a^i(s)\big)$ in *VAList* with wealth level $w \in \big(w^i(s), w^{i+1}(s)\big]$.

The backward-induction method uses the inductive foundation from the previous section and thus sets $w^0(s) = -\infty$, $v^0_\ell(s) = v_\ell^{\pi^{**}_{\mathsf{e}}}(s)$, $v^0_{\mathsf{e}}(s) = v^*_{\mathsf{e}}(s)$ and $a^0(s) = \pi^{**}_{\mathsf{e}}(s)$ for all states $s \in S'$ on Lines 1–4. It also initializes the priority queue *PQ*. The wealth level threshold $\underline{w}$ is the first non-infinity wealth level w^i removed from the priority queue in the main loop. The backward-induction method then proceeds in a backward fashion. Consider any state $s \in S'$ and any segment $\big(w^i(s), w^{i+1}(s)\big]$, where the wealth level $w^{i+1}(s)$ is still to be determined. We demand that two properties hold for this segment: First, the same action $a^i(s)$ has to be $\mathsf{MEU}_{1\mathsf{s}}$-optimal for all wealth levels $w \in \big(w^i(s), w^{i+1}(s)\big]$ (Property 1). Second, for all wealth levels $w \in \big(w^i(s), w^{i+1}(s)\big]$, actions $a \in A_s$ and states $s' \in S$ with $P(s'|s,a) > 0$, there exists a $j^{s,i}_{a,s'}$ such that $w^{j^{s,i}_{a,s'}+1}(s') \le w^i(s)$ and $w + r(s,a,s') \in \big(w^{j^{s,i}_{a,s'}}(s'), w^{j^{s,i}_{a,s'}+1}(s')\big]$, that is, all possible wealth levels after the action execution should be in the same segment (Property 2). Therefore, we have for all wealth levels $w \in \big(w^i(s), w^{i+1}(s)\big]$ according to Eq. (10) and Eq. (11)

$$\begin{aligned} v^*_{1\mathsf{s}}(s,w) &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) v^*_{1\mathsf{s}}(s', w + r(s,a,s')) \\ &= \max_{a \in A_s} \sum_{s' \in S} P(s'|s,a) \Big[w + r(s,a,s') + v_\ell^{j^{s,i}_{a,s'}}(s') \\ &\qquad + D\gamma^{w + r(s,a,s')} v_{\mathsf{e}}^{j^{s,i}_{a,s'}}(s') \Big] \end{aligned}$$

Algorithm 1 Backward-Induction Method

We define $\mathrm{succ}(s,a) = \{s' \in S \,|\, a \in A_s, P(s'|s,a) > 0\}$ and $\mathrm{pred}(s,a) = \{s' \in S \,|\, a \in A_{s'}, P(s|s',a) > 0\}$. We use the following operations on priority queues: Insert(*PQ*, s, w) inserts s into priority queue *PQ* with key w, IsMember(*PQ*, s) tests whether s is in *PQ*, GetKey(*PQ*, s) returns the key of s in *PQ* (s needs to be in *PQ*), DecreaseKey(*PQ*, s, w) decreases the current key of s in *PQ* to w (s needs to be in *PQ* with a key greater than w), and ExtractMin(*PQ*) removes a state with the lowest key from *PQ* and returns both the state and its key.

VAList = BackwardInductionOneSwitch($S, \{A_s\}, P, r, D, \gamma, w_0$)

1: determine v^*_{e}, $v_\ell^{\pi^{**}_{\mathsf{e}}}$, and π^{**}_{e};
2: **for all** $s \in S'$ **do**
3: AddList(*VAList*, $s, -\infty, v_\ell^{\pi^{**}_{\mathsf{e}}}(s), v^*_{\mathsf{e}}(s), \pi^{**}_{\mathsf{e}}(s)$);
4: Insert(*PQ*, $s, -\infty$);
5: $A \leftarrow \bigcup_{s \in S'} A_s$;
6: **while** ¬IsEmpty(*PQ*) **do**
7: $s, w^i \leftarrow$ ExtractMin(*PQ*);
8: $v^i_{1\mathsf{s}} \leftarrow -\infty$;
9: **for all** $a \in A_s$ **do**
10: $q^i_\ell(s,a), q^i_{\mathsf{e}}(s,a) \leftarrow 0$;
11: **for all** $s' \in \mathrm{succ}(s,a)$ **do**
12: $v^j_\ell(s'), v^j_{\mathsf{e}}(s') \leftarrow$ GetValues(*VAList*, $s', w^i + r(s,a,s')$);
13: $q^i_\ell(s,a) \leftarrow q^i_\ell(s,a) + P(s'|s,a)\Big[r(s,a,s') + v^j_\ell(s')\Big]$;
14: $q^i_{\mathsf{e}}(s,a) \leftarrow q^i_{\mathsf{e}}(s,a) + P(s'|s,a)\gamma^{r(s,a,s')} v^j_{\mathsf{e}}(s')$;
15: **if** $w^i \ne -\infty$ **then**
16: $q^i_{1\mathsf{s}}(s,a) \leftarrow w^i + q^i_\ell(s,a) + D\gamma^{w^i} q^i_{\mathsf{e}}(s,a)$;
17: **if** $q^i_{1\mathsf{s}}(s,a) > v^i_{1\mathsf{s}}$ **or** $\big(q^i_{1\mathsf{s}}(s,a) = v^i_{1\mathsf{s}}$ **and** $q^i_{\mathsf{e}}(s,a) < v^i_{\mathsf{e}}\big)$ **then**
18: $v^i_{1\mathsf{s}} \leftarrow q^i_{1\mathsf{s}}(s,a)$;
19: $a^i, v^i_\ell, v^i_{\mathsf{e}} \leftarrow a, q^i_\ell(s,a), q^i_{\mathsf{e}}(s,a)$;
20: **if** $w^i \ne -\infty$ **then**
21: AddList(*VAList*, $s, w^i, v^i_\ell, v^i_{\mathsf{e}}, a^i$);
22: **for all** $a \in A_s$ **do**
23: InsertSP(*PQ*, $s, w^i, v^i_\ell, v^i_{\mathsf{e}}, q^i_\ell(s,a), q^i_{\mathsf{e}}(s,a), \gamma, w_0$);
24: **if** $w^i \ne -\infty$ **then**
25: **for all** $a \in A$ **do**
26: **for all** $s' \in \mathrm{pred}(s,a)$ **do**
27: InsertNeg(*PQ*, $s', w^i - r(s',a,s), w_0$);

InsertSP(*PQ*, $s, w, v_\ell, v_{\mathsf{e}}, q_\ell, q_{\mathsf{e}}, \gamma, w_0$)

1: **if** $v_{\mathsf{e}} \ne q_{\mathsf{e}}$ **then**
2: $tmp \leftarrow \dfrac{q_\ell - v_\ell}{v_{\mathsf{e}} - q_{\mathsf{e}}}$;
3: **if** $tmp > D\gamma^{w_0}$ **then**
4: $\hat{w} \leftarrow \log_\gamma\big(\frac{1}{D} \cdot tmp\big)$;
5: **if** $\hat{w} > w$ **then**
6: InsertNeg(*PQ*, $s, \hat{w}, w_0$);

InsertNeg(*PQ*, s, w, w_0)

1: **if** $w < w_0$ **then**
2: **if** IsMember(*PQ*, s) **then**
3: **if** GetKey(*PQ*, s) $> w$ **then**
4: DecreaseKey(*PQ*, s, w);
5: **else**
6: Insert(*PQ*, s, w);

$$= \max_{a \in A_s} \left[w + \sum_{s' \in S} P(s'|s,a) \left[r(s,a,s') + v_\ell^{j^{s,i}_{a,s'}}(s') \right] + D\gamma^w \sum_{s' \in S} P(s'|s,a) \gamma^{r(s,a,s')} v_{\mathsf{e}}^{j^{s,i}_{a,s'}}(s') \right]$$
$$= \max_{a \in A_s} \left[w + q_\ell^i(s,a) + D\gamma^w q_{\mathsf{e}}^i(s,a) \right], \quad (13)$$

where we defined the values $q_\ell^i(s,a)$ and $q_{\mathsf{e}}^i(s,a)$ in the last step for use below.

First, the backward-induction method calculates action $a^i(s)$ on Lines 8–21. Since the optimal value function $v_{1\mathrm{s}}^*(s,\cdot)$ is continuous in the wealth level w and action $a^i(s)$ is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for all wealth levels $w \in \left(w^i(s), w^{i+1}(s)\right]$, it is also $\mathsf{MEU}_{1\mathrm{s}}$-optimal for wealth level $w^i(s)$, which implies according to Eq. (13)

$$a^i(s) \in \hat{A}(s) = \arg\max_{a \in A_s} \left[w^i(s) + q_\ell^i(s,a) + D\gamma^{w^i(s)} q_{\mathsf{e}}^i(s,a) \right],$$

where we defined the action sets $\hat{A}(s)$ for use below. The action sets $\hat{A}(s)$ can contain actions that are not $\mathsf{MEU}_{1\mathrm{s}}$-optimal for all wealth levels $w \in \left(w^i(s), w^{i+1}(s)\right]$. Figure 3(b) illustrates such actions. The action set $\hat{A}(s^0)$ includes actions 2, 3, 6 (the blue, black, and cyan graphs, respectively) for the segment $\left(w^i(s^0), w^{i+1}(s^0)\right]$, all of which intersect at the same point. Only action 2 is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for this segment, action 3 is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for the previous segment, and action 6 is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for neither segment. In general, the spurious actions can be eliminated by comparing the actions in the action set $\hat{A}(s)$ for wealth level $w^i(s) + \delta$, where δ is a positive infinitesimal. Consider any actions $a, a' \in \hat{A}(s)$. Then, we have

$$w^i(s) + q_\ell^i(s,a) + D\gamma^{w^i(s)} q_{\mathsf{e}}^i(s,a) = w^i(s) + q_\ell^i(s,a') + D\gamma^{w^i(s)} q_{\mathsf{e}}^i(s,a'). \quad (14)$$

Assume that action a is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for all wealth levels $w \in \left(w^i(s), w^{i+1}(s)\right]$, whereas action a' is not. Then, action a is also $\mathsf{MEU}_{1\mathrm{s}}$-optimal for wealth level $w^i(s) + \delta$, whereas action a' is not. Thus, we have

$$w^i(s) + \delta + q_\ell^i(s,a) + D\gamma^{w^i(s)+\delta} q_{\mathsf{e}}^i(s,a) > w^i(s) + \delta + q_\ell^i(s,a') + D\gamma^{w^i(s)+\delta} q_{\mathsf{e}}^i(s,a').$$

By subtracting Eq. (14) from the inequality, we have

$$\delta + D\gamma^{w^i(s)} \left[\gamma^\delta - 1\right] q_{\mathsf{e}}^i(s,a) > \delta + D\gamma^{w^i(s)} \left[\gamma^\delta - 1\right] q_{\mathsf{e}}^i(s,a')$$
$$q_{\mathsf{e}}^i(s,a) < q_{\mathsf{e}}^i(s,a'),$$

Thus, the backward-induction method chooses any action $a^i(s) \in \arg\min_{a \in \hat{A}(s)} q_{\mathsf{e}}^i(s,a)$.

Second, the backward-induction method calculates the values $v_\ell^i(s) = q_\ell^i(s,a^i(s))$ and $v_{\mathsf{e}}^i(s) = q_{\mathsf{e}}^i(s,a^i(s))$ on Lines 8–21 since $v_{1\mathrm{s}}^*(s,w) = w + q_\ell^i(s,a^i(s)) + D\gamma^w q_{\mathsf{e}}^i(s,a^i(s))$ according to Eq. (13) and $v_{1\mathrm{s}}^*(s,w) = w + v_\ell^i(s) + D\gamma^w v_{\mathsf{e}}^i(s)$ according to our representation of the piecewise one-switch functions for all wealth levels $w \in (w^i(s), w^{i+1}(s)]$.

Third, the backward-induction method calculates the wealth levels $w^{i+1}(s)$. Property 1 implies that action $a^i(s)$ is $\mathsf{MEU}_{1\mathrm{s}}$-optimal for wealth level $w^{i+1}(s)$. We thus have for all actions $a \in A_s$

$$w^{i+1}(s) + q_\ell^i(s,a^i(s)) + D\gamma^{w^{i+1}(s)} q_{\mathsf{e}}^i(s,a^i(s)) \geq w^{i+1}(s) + q_\ell^i(s,a) + D\gamma^{w^{i+1}(s)} q_{\mathsf{e}}^i(s,a)$$
$$q_\ell^i(s,a^i(s)) + D\gamma^{w^{i+1}(s)} q_{\mathsf{e}}^i(s,a^i(s)) \geq q_\ell^i(s,a) + D\gamma^{w^{i+1}(s)} q_{\mathsf{e}}^i(s,a)$$
$$v_\ell^i(s) + D\gamma^{w^{i+1}(s)} v_{\mathsf{e}}^i(s) \geq q_\ell^i(s,a) + D\gamma^{w^{i+1}(s)} q_{\mathsf{e}}^i(s,a)$$
$$w^{i+1}(s) \leq \log_\gamma \left(\frac{1}{D} \cdot \frac{q_\ell^i(s,a) - v_\ell^i(s)}{v_{\mathsf{e}}^i(s) - q_{\mathsf{e}}^i(s,a)} \right),$$

where the right-hand side of the last inequality defines a potential switching point for the $\mathsf{MEU}_{1\mathrm{s}}$-optimal action and thus a potential value for the wealth level $w^{i+1}(s)$ if the argument of the logarithm is positive and the logarithm thus is defined, as illustrated in Figure 3(b). The determination of these potential switching points is essentially the same as the determination of the wealth level threshold (with the only difference being whether $w^i = -\infty$), which allows the backward-induction method to combine their calculation in the main loop on Lines 22–23 using InsertSP. Property 2 implies that $w + r(s,a,s') \leq w^{j^{s,i}_{a,s'}+1}(s')$ and thus that $w \leq w^{j^{s,i}_{a,s'}+1}(s') - r(s,a,s')$, where the right-hand side of this inequality defines another potential switching point for the $\mathsf{MEU}_{1\mathrm{s}}$-optimal action and thus another potential value for the wealth level $w^{i+1}(s)$, as illustrated in Figure 3(c) where wealth level w^i takes the role of wealth level $w^{j^{s,i}_{a,s'}+1}$ and reward c_k takes the role of reward $r(s,a,s')$. The backward-induction method calculates the potential switching points on Lines 25–27 using InsertNeg.

The backward-induction method stores the potential switching points in the priority queue *PQ* and processes them in order of increasing wealth levels since the values of segment $(w^i(s), w^{i+1}(s)]$ depend on the values of segments $(w^j(s'), w^{j+1}(s')]$ for one or more j with $w^{j+1}(s') \leq w^i(s)$, which need to have been calculated already. The backward-induction method terminates when the priority queue *PQ* is empty, which happens in a finite amount of time since the wealth level threshold $\underline{w}$ is a finite negative value and the values of the processed switching points are monotonically increasing by at least a positive constant and guaranteed to be non-positive [7]. Thus, the $\mathsf{MEU}_{1\mathrm{s}}$-optimal value functions are indeed piecewise one-switch functions with a finite number of segments. For the termite GDMDP, the backward-induction method finds the following $\mathsf{MEU}_{1\mathrm{s}}$-optimal policy for the initial wealth level $w_0 = 0$:

$$\pi_{1\mathrm{s}}^*(s^0, w) = \begin{cases} 1 & w \in (-316.4, 0] \\ 2 & w \in (-1482.0, -316.4] \\ 3 & w \in (-\infty, -1482.0]. \end{cases}$$

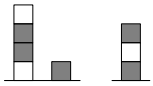


Figure 4: Painted Blocksworld Problem

7. EXAMPLE

We use the painted blocksworld problem from [9] to illustrate risk-sensitive planning with one-switch utility functions. The domain is a standard blocksworld domain with five blocks that are either white (W) or black (B). However, the move action succeeds only with probability 0.5. When it fails, the block drops directly onto the table. One can also execute a paint action that changes the color of any one block and always succeeds. The move action (M) has a reward of -1, and the paint action (P) has a reward of -3. Figure 4(a) shows the initial state. The goal is to build a stack of three blocks as shown in Figure 4(b). The painted

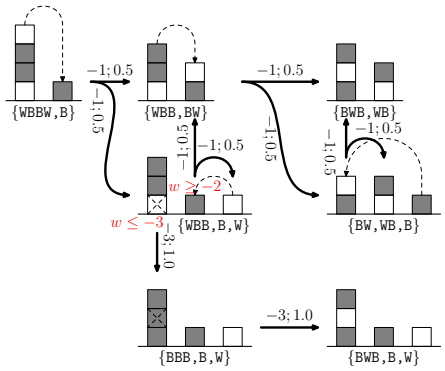


Figure 5: $\mathsf{MEU}(U_{1s})$-Optimal Policy

{WBB,BW} and {BW,WB,B}

$(w^i(s), v^i_{1s}(s))$	$v^i_\ell(s)$	$v^i_e(s)$	$a^i(s)$
$(0.00, -4.50)$			
$(-\infty, -\infty)$	-2.00	-5.00	M

{BBB,B,W}

$(w^i(s), v^i_{1s}(s))$	$v^i_\ell(s)$	$v^i_e(s)$	$a^i(s)$
$(0.00, -5.31)$			
$(-\infty, -\infty)$	-3.00	-4.63	P

{WBBW,B}

$(w^i(s), v^i_{1s}(s))$	$v^i_\ell(s)$	$v^i_e(s)$	$a^i(s)$
$(0.00, -15.72)$			
$(-0.38, -18.52)$	-4.25	-22.94	M
$(-1.38, -28.61)$	-4.50	-22.52	M
$(-\infty, -\infty)$	-5.00	-22.03	M

{WBB,B,W}

$(w^i(s), v^i_{1s}(s))$	$v^i_\ell(s)$	$v^i_e(s)$	$a^i(s)$
$(0.00, -15.72)$			
$(-0.38, -18.52)$	-4.25	-22.94	M
$(-1.38, -28.61)$	-4.50	-22.52	M
$(-2.38, -44.43)$	-5.00	-22.03	M
$(-\infty, -\infty)$	-6.00	-21.43	P

Figure 6: MEU_{1s}-Optimal Value Functions

blocksworld problem has 162 states, which we describe as sets of stacks by listing the blocks in each stack from bottom to top. For example, the initial state is {WBBW,B}. We use the backward-induction method to find an MEU_{1s}-optimal policy for the one-switch utility function $U_{1s}(w) = w - 0.5 \times 0.6^w$. Figure 5 depicts this MEU_{1s}-optimal policy, and Figure 6 shows the MEU_{1s}-optimal value functions in the form of *VAList* entries for the five non-goal states that are reachable from the initial state if the agent follows the MEU_{1s}-optimal policy, where we use the shorthand notation $v^i_{1s}(s) = v^*_{1s}(s, w^i(s))$ and also include the values for the case where wealth level $w^i(s)$ is equal to the initial wealth level $w_0 = 0$, which do not appear in *VAList*. The MEU_{1s}-optimal action in state {WBB,B,W} depends on the wealth level. It is a move action for wealth level -1 or -2 and a paint action for wealth level -3. It takes the backward-induction method only about 0.45 seconds to obtain the MEU_e-optimal policy on a Dell Latitude D600 laptop, while it takes functional value iteration about 1.45 seconds, although our backward-induction method in general is not guaranteed to be faster than functional value iteration and can, in fact, be slower. A more thorough study of the running times of functional value iteration and the backward induction method is underway.

8. CONCLUSIONS

The backward-induction method exploits the structure of one-switch utility functions to determine maximal expected utilities for given GDMDPs and one-switch utility functions in finite time, different from the previous functional value iteration method that typically determines only approximately maximal expected utilities. In the future, we intend to study how to exploit the structure of other nonlinear utility functions in similar ways and develop methods that scale up to even larger planning problems.

9. REFERENCES

[1] D. E. Bell. One-switch utility functions and a measure of risk. *Management Science*, 34(12):1416–1424, 1988.

[2] D. E. Bell and P. C. Fishburn. Strong one-switch utility. *Management Science*, 47(4):601–604, 2001.

[3] D. P. Bertsekas and J. N. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

[4] J. L. Corner and P. D. Corner. Characteristics of decisions in decision analysis practice. *The Journal of Operational Research Society*, 46:304–314, 1995.

[5] G. M. Gelles and D. W. Mitchell. Broadly decreasing risk aversion. *Management Science*, 45:1432–1439, 1999.

[6] S. Koenig and R. G. Simmons. 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, 1994.

[7] Y. Liu. Decision-Theoretic Planning under Risk-Sensitive Planning Objectives. Dissertation, College of Computing, Georgia Institute of Technology, 2005.

[8] Y. Liu and S. Koenig. Existence and finiteness conditions for risk-sensitive planning: Results and conjectures. In *Proceedings of the Twentieth Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, 2005.

[9] Y. Liu and S. Koenig. Risk-sensitive planning with one-switch utility functions: Value iteration. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pages 993–999, 2005.

[10] C. McMillen and M. Veloso. Thresholded rewards: Acting optimally in timed, zero-sum games. In *Proceedings of The Twenty-Second Conference on Artificial Intelligence (AAAI-07)*, 2007.

[11] Y. Nakamura. Sumex utility functions. *Mathematical Social Sciences*, 31:39–47, 1996.

[12] E. Nikolova, M. Brand, and D. R. Karger. Optimal route planning under uncertainty. In *Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling (ICAPS-06)*, 2006.

[13] S. D. Patek. On terminating Markov decision processes with a risk averse objective function. *Automatica*, 37(9):1379–1386, 2001.

[14] P. Perny, O. Spanjaard, and L.-X. Storme. State space search for risk-averse agents. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2353–2358, 2007.

[15] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[16] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1st edition, 1944.