Planning-Task Transformations for Soft Deadlines

Sven Koenig

College of Computing, Georgia Institute of Technology
Atlanta, GA 30305, USA
skoenig@cc.gatech.edu

Abstract. Agents often have preference models that are more complicated than minimizing the expected execution cost. In this paper, we study how they should act in the presence of uncertainty and immediate soft deadlines. Delivery robots, for example, are agents that are often confronted with immediate soft deadlines. We introduce the additive and multiplicative planning-task transformations, that are fast representation changes that transform planning tasks with convex exponential utility functions to planning tasks that can be solved with variants of standard deterministic or probabilistic artificial intelligence planners. Advantages of our representation changes include that they are context-insensitive, fast, scale well, allow for optimal and near-optimal planning, and are grounded in utility theory. Thus, while representation changes are often used to make planning more efficient, we use them to extend the functionality of existing planners, resulting in agents with more realistic preference models.

1 Introduction

To determine how agents should act, one has to know their preference model. This is an often neglected research topic in agent theory and artificial intelligence in general. Many artificial intelligence planners, for example, assume that agents want to minimize the expected cost or maximize the expected reward. Unfortunately, agents often have preference models that are more complicated than that. They have, for example, to trade-off between different resource consumptions (such as money, energy, and time), take risk attitudes in high-stake decision situations into account, or act in the presence of deadlines. Our research program aims at building agents with these preference models. In this paper, we study planning tasks for delivery robots in the presence of uncertainty and immediate soft deadlines. Uncertainty can be caused by actuator and sensor noise, limited sensor ranges, map uncertainty, uncertainty about the initial pose of the robot, and uncertainty about the dynamic state of the environment (such as people opening and closing doors or moving furniture around). Immediate soft deadlines are caused by delivery requests that are not made in advance but the goods are needed right away and their utility declines over time, for example because the goods become obsolete or lose their value over time (coffee, for example, gets cold). While artificial intelligence planners usually determine plans that achieve the goal with minimal expected execution time, here they have to determine plans that achieve the goal with maximal expected utility for the convex exponential utility functions that model the immediate soft deadlines. Maximizing expected utility and minimizing expected execution time

result in the same plans if either the domain is deterministic or the utility function is linear, but these properties rarely hold. The goal of this paper is to develop a planning methodology for planning in the presence of uncertainty and immediate soft deadlines that determines optimal or near-optimal plans for agents and scales up to large domains. To this end, we combine constructive planning approaches from artificial intelligence with more descriptive approaches from utility theory [1, 18], utilizing known properties of exponential utility functions [17, 10]. In particular, we develop efficient representation changes that transform planning tasks with convex exponential utility functions to planning tasks that can be solved with variants of standard deterministic or probabilistic artificial intelligence planners such as [5, 3, 8, 7, 13, 2, 16]. Our planning-task transformations provide alternatives to the few existing approaches to planning with non-linear utility functions, namely approximate planners [9], planners with limited look-ahead [11], intelligent branch-and-bound methods [19], and hill-climbing methods [15]. Advantages of our planning-task transformations include that they are simple context-insensitive representation changes that are fast, scale well, allow for optimal and near-optimal planning, and are grounded in utility theory. While we use delivery robots to illustrate the problem and our solution, our insights apply to other agents as well, including agents that have to make decisions in high-stake decision situations.

2 Immediate Soft Deadlines

Delivery robots are agents that are often confronted with deadlines [6]. Immediate deadlines coincide with the time at which the execution of a plan begins. Soft deadlines are those whose utility does not drop to zero immediately after the deadline has passed but rather declines slowly. Immediate soft deadlines can often be modeled either exactly or approximately with convex exponential utility functions, where a utility function maps rewards r (here: the negative of the execution times) to the resulting real-valued utilities $u(r)$. Convex exponential utility functions are of the form $u(r) = \gamma^r$ with parameter $\gamma > 1$ or linear transformations thereof. The smaller γ, the softer the deadline. An example is the utility function $u(r) = (2^{1/300})^r$, where the reward r is the negative of the execution time, measured in seconds. In this particular case, the utility of the delivery halves every five minutes. A simple example of an immediate soft deadline occurs when coffee is delivered. Coffee gets colder during the delivery and one therefore often wants to maximize its expected temperature at the time of delivery. The utility function expresses how the temperature of the coffee depends on the execution time. Since the rate of cooling is proportional to the temperature difference between the cup and its environment, the utility function is convex exponential in the negative of the execution time. A more complex example of an immediate soft deadline occurs when printouts are delivered on demand. Imagine, for example, that you are debugging a program on your computer. To get a better overview of the program, you print it out and send your office delivery robot to fetch the printout from the remote printer room. In this case, the printout is needed right away, but you do not need it any longer once you determine the problem with the program. The probability that you have not found the problem decreases during the delivery and one therefore often wants to maximize the expected probability that the problem has not been found at the time of delivery. The utility func-

tion expresses how this probability depends on the execution time. If the probability that the problem is found in any period of time Δt is p, then the probability that it has not been found after n such time periods (assuming probabilistic independence) is $(1-p)^n$. Thus, the utility function is convex exponential in the negative of the execution time, just like in the coffee-delivery example.

3 Utility Theory

In this paper, we make use of the following concepts from utility theory [1, 18]: If the execution of some plan $plan$ leads with probability p_i to reward r_i, then its expected reward is $er(plan) := \sum_i [p_i r_i]$ and its expected utility is $eu(plan) := \sum_i [p_i u(r_i)]$ for the convex exponential utility function $u(r) = \gamma^r$ with parameter $\gamma > 1$ that models the immediate soft deadline. (If the utility function that models the immediate soft deadline is a linear transformation of this utility function, we can use this utility function instead.) The certainty equivalent of the plan is $ce(plan) := u^{-1}(eu(plan))$. A decision maker is indifferent between a plan and a deterministic plan (that is, obtaining a certain reward for sure) if and only if the reward of the deterministic plan is the same as the certainty equivalent of the other plan. This explains the name of this concept from utility theory. It is easy to see that maximizing the certainty equivalent and maximizing the expected utility are equivalent.

4 Approximation Error

Traditional artificial intelligence planners determine plans that achieve the goal with minimal expected execution time or, equivalently, with maximal expected reward. Our planners, on the other hand, have to determine plans that achieve the goal with maximal expected utility. We first show that, in general, plans that achieve the goal with maximal expected reward cannot be used to closely approximate plans that achieve the goal with maximal expected utility. To determine the approximation error of choosing a plan with maximal expected reward over a plan with maximal expected utility, let $plan_{eu}$ denote a plan that achieves the goal with maximal expected utility and $plan_{er}$ a plan that achieves the goal with maximal expected reward. We quantify the approximation error err of choosing $plan_{er}$ over $plan_{eu}$ as the amount of time by which we have to delay the execution of $plan_{eu}$ so that the resulting expected utility equals the expected utility of $plan_{er}$. This is the idle time that is available for other tasks. Thus, if the execution of plan $plan_{eu}$ leads with probability p_i to reward r_i, then the approximation error err satisfies the equation

$$eu(plan_{er}) = \sum_i p_i u(r_i - err). \tag{1}$$

It follows that

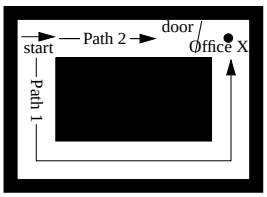

Fig. 1. A Simple Planning Example

$$
\begin{aligned}
ce(plan_{er}) &= u^{-1}(eu(plan_{er})) \\
&= u^{-1}(\sum_i p_i u(r_i - err)) \qquad \text{see (1)} \\
&= \log_\gamma(\sum_i p_i \gamma^{r_i - err}) \\
&= -err + \log_\gamma(\sum_i p_i \gamma^{r_i}) \\
&= -err + u^{-1}(\sum_i p_i u(r_i)) \\
&= -err + u^{-1}(eu(plan_{eu})) \\
&= -err + ce(plan_{eu}). \qquad (2)
\end{aligned}
$$

Consequently, the approximation error is $err = ce(plan_{eu}) - ce(plan_{er}) \geq 0$. To construct an example with a large approximation error, assume that the utility function is $u(r) = (2^{1/300})^r$ (as before). Consider a plan $plan_{bad}$ that leads with probability 0.37 to an execution time of 80.00 seconds and with the complementary probability to an execution time of 800.00 seconds. The expected utility of $plan_{bad}$ is $eu(plan_{bad}) = 0.41$, its certainty equivalent is $ce(plan_{bad}) = -389.31$ seconds, and its expected reward is $er(plan_{bad}) = -533.60$ seconds. Also consider a plan $plan_{worse}$ that leads with probability 1.00 to an execution time of 533.60 seconds. The expected utility of $plan_{worse}$ is $eu(plan_{worse}) = 0.29$, its certainty equivalent is $ce(plan_{worse}) = -533.60$ seconds, and its expected reward is $er(plan_{worse}) = -533.60$ seconds as well. The approximation error of choosing $plan_{worse}$ over $plan_{bad}$ calculates to roughly 2.5 minutes (144.29 seconds) for this delivery task whose expected execution time is only about 9 minutes (533.60 seconds). This example might appear to be constructed but delivery robots face indeed similar planning tasks. Assume, for example, that a robot operates in the corridor environment from Figure 1 and has to reach office X, at which point execution stops. There are only two candidate plans that are not clearly dominated by some other plan. The robot can either follow path 1 or path 2. Path 1 is known not to be blocked and thus results in a deterministic execution time, just like $plan_{worse}$ in the worst-case example above. Path 2 leads through a door that the robot is not able to open. Furthermore, the robot cannot sense whether the door is open or closed until it is close to the door. Thus, it has to move towards the door and, if the door is closed, return to

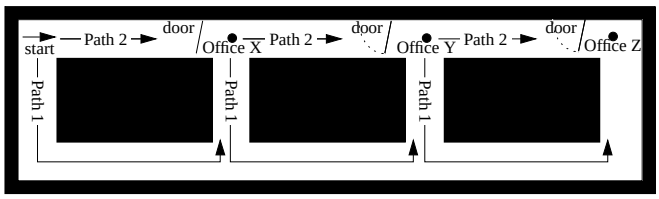

Fig. 2. Another Simple Planning Example

its start location and then take the long path to office X. Consequently, path 2 can result in different execution times depending on whether the door is open or closed, just like $plan_{bad}$ in the worst-case example above.

5 Optimal Planning

The previous section showed that it is important to directly determine plans that achieve the goal with maximal expected utility. Unfortunately, utility theory is a purely descriptive theory that specifies only what optimal plans are, but not how plans that achieve the goal with maximal expected utility can be obtained other than by enumerating every chronicle of every possible plan, where a chronicle is a specification of the state of the world over time, representing one possible course of execution of the plan. Operations research and control theory have picked up on the results from utility theory and use dynamic programming methods to determine plans that achieve the goal with maximal expected utility in the context of risk-sensitive Markov decision process models [14] and risk-sensitive control [20]. These methods do not utilize available domain knowledge. Artificial intelligence has investigated knowledge-based planners that promise to scale up to larger domains, but traditionally do not determine plans that achieve the goal with maximal expected utility. In the following, we therefore extend the range of planners that determine plans that achieve the goal with maximal expected utility for convex exponential utility functions from methods of operations research to variants of standard deterministic or probabilistic artificial intelligence planners. We limit our attention to plans that are mappings from states to actions and planning tasks where the chronicles of all plans have a bounded reward. In other words, they always terminate after a bounded execution time. We also assume that all rewards are negative, which is the case if the rewards correspond to resource consumptions.

5.1 The Additive Planning-Task Transformation

The additive planning-task transformation applies to special cases of planning tasks, namely those for which every action is deterministic in the sense that its execution always ends in the same state although it can result in different rewards. This requirement is similar to assumptions sometimes made in the literature [19]. For example, the plans that we studied in the context of the example from Figure 1 satisfy it because the actions correspond to plans (or "macro actions") for getting from one office to another and

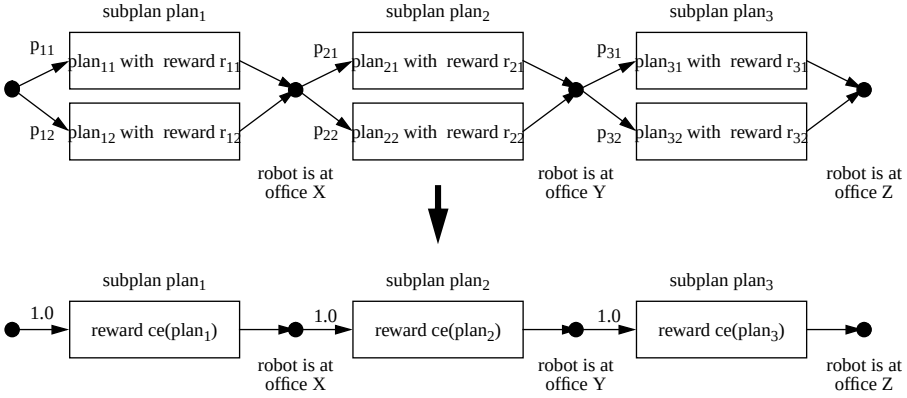

Fig. 3. Additive Planning-Task Transformation

move the robot to its destination, although the robot can take different paths depending on whether the door is open or closed and thus incur different execution times. The additive planning-task transformation converts the planning tasks by modifying all of their actions (everything else remains the same): If an action can be executed in state s and its execution leads with probability p_i and reward r_i to state s' (for all i), then it is replaced with a deterministic action. This action can be executed in state s and its execution leads with probability one and reward $u^{-1}(\sum_i [p_i u(r_i)])$ (= the certainty equivalent of the original action) to state s'. We illustrate the additive planning-task transformation using the planning task from Figure 2. This example is the simplest example we could devise that allows us to demonstrate the advantages of the additive planning-task transformation. It is similar to the example from Figure 1 but consists of three parts. The robot has to first visit office X to pick up a form, then obtain a signature in office Y, and finally deliver the signed form to office Z, at which point plan execution stops. We assume throughout this paper that the probabilities with which doors are open when the robot visits them are independent (even if the robot visits the same door multiple times). Consider the action that corresponds to trying to take path 2 to get from office X to office Y. If the robot reaches office Y in 120.00 seconds with probability 0.50 (= the door is open) and in 576.00 seconds with probability 0.50 (= the door is closed), then the action is replaced with a deterministic action that can be executed when the robot is at office X and whose execution moves the robot with probability one and reward $u^{-1}(0.50\,u(-120.00) + 0.50\,u(-576.00)) = \log_\gamma(0.50\,\gamma^{-120.00} + 0.50\,\gamma^{-576.00})$ to office Y. The other actions are transformed similarly.

We now explain why the additive planning-task transformation is such that a plan that achieves the goal with maximal reward for the transformed planning task corresponds to a plan that achieves the goal with maximal expected utility for the original planning task, see Figure 4. Exponential utility functions satisfy the Markov property, which implies that it is unimportant how much reward has already been accumulated for acting optimally in the future. This property of exponential utility functions is also known as the delta property [10] or, equivalently, constant local risk aversion [17].

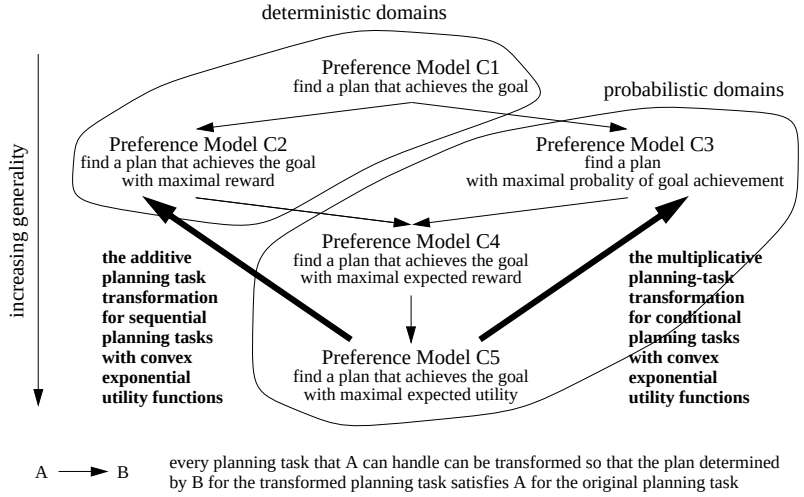

Fig. 4. Preference Models for Planning Tasks

Now consider the certainty equivalent of any sequential plan for the original planning task. As an example, we use the plan from Figure 3 (top) that solves the planning task from Figure 2 by trying to take path 2 for all three parts of the planning task. We use the following notation: The first action $plan_1$ corresponds to reaching office X. It has two possible subchronicles $plan_{11}$ and $plan_{12}$. Subchronicle $plan_{11}$ denotes the case where the door is open, whereas subchronicle $plan_{12}$ corresponds to the door being closed. Subchronicle $plan_{11}$ has reward r_{11} and occurs with probability p_{11}, subchronicle $plan_{12}$ has reward r_{12} and occurs with probability p_{12} (where $p_{11} + p_{12} = 1$), and similarly for the other actions and subchronicles. Then, the certainty equivalent of plan $plan_1 \cdot plan_2 \cdot plan_3$ (the concatenation of the three actions) is

$$
\begin{aligned}
& ce(plan_1 \cdot plan_2 \cdot plan_3) \\
& = u^{-1}(eu(plan_1 \cdot plan_2 \cdot plan_3)) \\
& = u^{-1}(\sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}[p_{1i}p_{2j}p_{3k}u(r_{1i}+r_{2j}+r_{3k})]) \\
& = \log_\gamma \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2}[p_{1i}p_{2j}p_{3k}(\gamma^{r_{1i}+r_{2j}+r_{3k}})] \\
& = \log_\gamma \sum_{i=1}^{2}[p_{1i}(\gamma^{r_{1i}})] + \log_\gamma \sum_{j=1}^{2}[p_{2j}(\gamma^{r_{2j}})] + \log_\gamma \sum_{k=1}^{2}[p_{3k}(\gamma^{r_{3k}})] \\
& = u^{-1}(\sum_{i=1}^{2}[p_{1i}u(r_{1i})]) + u^{-1}(\sum_{j=1}^{2}[p_{2j}u(r_{2j})]) + u^{-1}(\sum_{k=1}^{2}[p_{3k}u(r_{3k})]) \\
& = u^{-1}(eu(plan_1)) + u^{-1}(eu(plan_2)) + u^{-1}(eu(plan_3))
\end{aligned}
$$

$$= ce(plan_1) + ce(plan_2) + ce(plan_3). \quad (3)$$

Instead of calculating its certainty equivalent directly, we can first transform the plan. Its structure remains unchanged, but all of its actions are transformed (as described above). The reward of the transformed plan is the same as the certainty equivalent of the original plan. Figure 3 (bottom), for example, shows the transformation of the plan from Figure 3 (top). The reward of the transformed plan is $ce(plan_1)+ce(plan_2)+ce(plan_3)$, which is also the certainty equivalent of the original plan according to Equation 3. To determine a plan that achieves the goal with maximal expected utility for the original planning task, one can transform all actions of the planning task. Since there is a one-to-one correspondence between the sequential plans of the original planning task and the sequential plans of the transformed planning task, a plan that achieves the goal with maximal reward for the transformed planning task corresponds to a plan that achieves the goal with maximal certainty equivalent (and thus also maximal expected utility) among all sequential plans for the original planning task. To summarize, the original planning task can be solved by applying the additive planning-task transformation and then solving the transformed planning task with any planner that determines plans that achieve the goal with maximal reward.

The additive planning-task transformation, although simple, can solve standard kinds of delivery tasks. We give two examples in the following. The first example is the planning task from Figure 3. It can be solved as follows:

$$\begin{aligned}
&\max_{plan_1, plan_2, plan_3} ce(plan_1 \cdot plan_2 \cdot plan_3) \\
&= \max_{plan_1, plan_2, plan_3} [ce(plan_1) + ce(plan_2) + ce(plan_3)] \quad \text{see (3)} \\
&= \max_{plan_1} ce(plan_1) + \max_{plan_2} ce(plan_2) + \max_{plan_3} ce(plan_3).
\end{aligned}$$

Thus, to determine a plan that achieves the goal with maximal expected utility for the original planning task, a planner can determine separate actions for the three parts of the planning task, each of which achieves its subgoal with maximal certainty equivalent and thus also with maximal expected utility. This can be done by transforming each part of the planning task individually with the additive planning-task transformation and solving the transformed planning task. The three resulting actions combined then form a plan that achieves the goal with maximal expected utility for the original planning task. This is no longer true in the next example, that is similar to the previous one. This time, however, the robot starts at the secretary's office with the task of collecting ten signatures on a form and returning it to the secretary. This planning task is essentially one of task sequencing. The additive planning-task transformation converts the planning task to a traveling-salesman problem on a directed graph with deterministic edge rewards. The reward of an edge is the certainty equivalent of a plan with maximal expected utility that moves from the office at the beginning of the edge to the office at its end. The traveling-salesman problem can then be solved with traveling-salesman problem methods or standard deterministic artificial intelligence planners. While these examples show that the additive planning-task transformation is useful, it cannot be

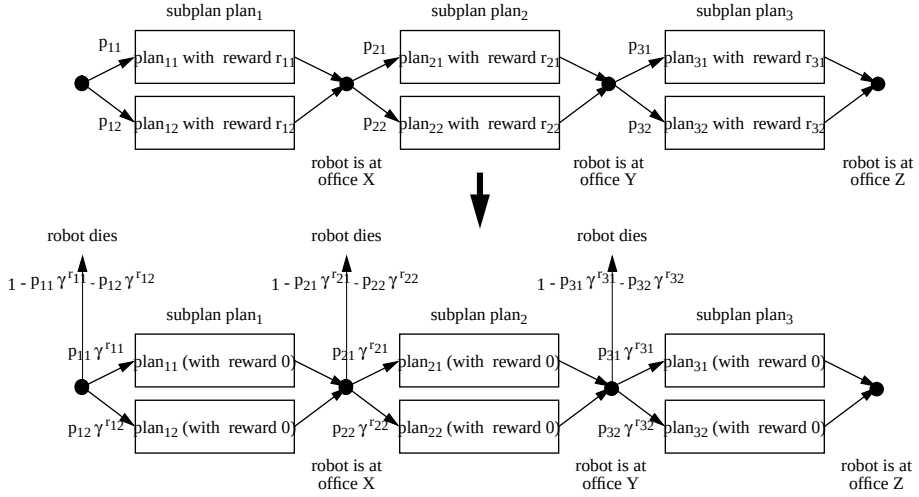

Fig. 5. Multiplicative Planning-Task Transformation

used to solve all planning tasks because it is often not the case that each action is atomic and always ends in the same state. As an example, consider again a robot that has to collect ten signatures. The actions move the robot to a specified office. They are not atomic in practice because their execution can be interrupted. For example, while the robot moves to some office, it might have to take a detour because a door is closed. If this detour leads past another office, it can be advantageous for the robot to go first to that office in order to obtain another signature on the way to its original destination. This suggests considering "move to the door" as an action because this would allow the robot to re-plan and change its destination when it recognizes that the door is closed. However, this action can end in two different states: the robot being at an open door or the robot being at a closed door. Thus, it does not satisfy the assumption of the additive planning-task transformation and we have to consider ways of determining conditional (rather than sequential) plans that achieve the goal with maximal expected utility.

5.2 The Multiplicative Planning-Task Transformation

The multiplicative planning-task transformation is more general than the additive planning-task transformation. It extends our previous work on planning with exponential utility functions [12] from reactive planning in high-stake decision situations to deliberative planning in the presence of immediate soft deadlines. The multiplicative planning-task transformation applies to planning tasks that can be solved optimally with conditional plans and converts them by modifying all of their actions (everything else remains the same): If an action can be executed in state s and its execution leads with probability p_i and reward r_i to state s_i (for all i), then it is replaced with an action that can be executed in state s and whose execution leads with probability $p_i\gamma^{r_i}$ to state s_i (for all i) and with probability $1 - \sum_i [p_i\gamma^{r_i}]$ to a new nongoal state ("death") in which execution stops. The rewards do not matter. We illustrate the multiplicative

planning-task transformation using the planning task from Figure 2 (as before). Consider the action that corresponds to trying to take path 2 to get from office X to office Y. If the robot reaches office Y in 120.00 seconds with probability 0.50 (= the door is open) and in 576.00 seconds with probability 0.50 (= the door is closed), then the action is replaced with an action that can be executed when the robot is at office X and whose execution moves the robot with probabilities $0.5\gamma^{-120.00}$ and $0.5\gamma^{-576.00}$ to office Y and with probability $1 - 0.5\gamma^{-120.00} - 0.5\gamma^{-576.00}$ to the new nongoal state "death." The other actions are transformed similarly.

We now explain why the multiplicative planning-task transformation is such that a plan that achieves the goal with maximal probability for the transformed planning task and always stops in the goal states or "death" corresponds to a plan that achieves the goal with maximal expected utility for the original planning task, see Figure 4. We first consider sequential plans as a special case and then conditional plans in general. As an example, we use again the sequential plan $plan_1 \cdot plan_2 \cdot plan_3$ from Figure 5 (top). It solves the planning task from Figure 2 by trying to take path 2 for all three parts of the planning task. Its expected utility for the original planning task is

$$\begin{aligned}
&eu(plan_1 \cdot plan_2 \cdot plan_3) \\
&= \sum_{i=1}^{2} \sum_{j=1}^{2} \sum_{k=1}^{2} [p_{1i} p_{2j} p_{3k} u(r_{1i} + r_{2j} + r_{3k})] \\
&= \sum_{i=1}^{2} \sum_{j=1}^{2} \sum_{k=1}^{2} [p_{1i} p_{2j} p_{3k} \gamma^{r_{1i} + r_{2j} + r_{3k}}] \\
&= \sum_{i=1}^{2} [p_{1i} \gamma^{r_{1i}}] \times \sum_{j=1}^{2} [p_{2j} \gamma^{r_{2j}}] \times \sum_{k=1}^{2} [p_{3k} \gamma^{r_{3k}}] \\
&= \sum_{i=1}^{2} \overline{p}_{1i} \times \sum_{j=1}^{2} \overline{p}_{2j} \times \sum_{k=1}^{2} \overline{p}_{3k}, \qquad (4)
\end{aligned}$$

where the parameters $\overline{p}_{mn}$ are new values with $\overline{p}_{mn} := p_{mn} \gamma^{r_{mn}}$. These values satisfy $0 \leq \overline{p}_{mn} \leq p_{mn}$ according to our assumption that the utility function is convex exponential ($\gamma > 1$) and all rewards are negative ($r_{mn} < 0$). Thus, the parameters $\overline{p}_{mn}$ can be interpreted as probabilities of not dying during the execution of the transformed action, one probability for each of its subchronicles. The complementary probability $1 - \sum_n \overline{p}_{mn} \geq 0$ is the probability of dying during the execution of the transformed action. Instead of calculating the expected utility of the original plan directly, we can first transform it. Its structure remains unchanged, but all of its actions are transformed (as described above). The expected utility of the original plan is the same as the probability of not dying while executing the transformed plan, which is the product of the probabilities of not dying while executing its actions (due to probabilistic independence). Figure 5 (bottom), for example, shows the transformation of the plan from Figure 5 (top). The probability of not dying during the execution of the transformed plan is $\sum_{i=1}^{2} [p_{1i} \gamma^{r_{1i}}] \times \sum_{j=1}^{2} [p_{2j} \gamma^{r_{2j}}] \times \sum_{k=1}^{2} [p_{3k} \gamma^{r_{3k}}]$, which is also the expected utility of the original plan according to Equation 4. Now consider an arbitrary conditional

plan for the original planning task. We can use the multiplicative planning-task transformation on the plan and it remains true that the expected utility of the original plan is the same as the probability of not dying while executing the transformed plan. This is so because the expected utility of the original plan is the sum of the utility contributions over all of its chronicles, where the utility contribution of a chronicle is the product of its probability and utility. A chronicle is a sequence of subchronicles. If subchronicle i of the chronicle has probability p_i and reward r_i, then the utility contribution of the chronicle is

$$(\prod_i p_i)\, u(\sum_i r_i) = (\prod_i p_i)\, \gamma^{\sum_i r_i} = \prod_i [p_i \gamma^{r_i}] = \prod_i \overline{p}_i. \tag{5}$$

A chronicle of the original plan corresponds to several chronicles of the transformed plan, only one of which does not end in "death." Equation 5 is the probability of this chronicle. The sum of these probabilities over all chronicles of the original plan is the probability of not dying during the execution of the transformed plan. To determine plans that achieve the goal with maximal expected utility for the original planning task one can transform all actions of the planning task. Then, a plan that achieves the goal with maximal probability for the transformed planning task and always stops in only the goal states or "death" also achieves the goal with maximal expected utility for the original planning task. This is so because there is a one-to-one correspondence between the plans of the transformed planning task that always stop in only the goal states or "death" and the plans that achieve the goal for the original planning task. The expected utility of a plan for the original planning task is the same as the probability of not dying during the execution of the corresponding plan for the transformed planning task, which is the same as the probability with which the transformed plan achieves the goal if it stops in only the goal states or "death." To summarize, the original planning task can be solved by applying the multiplicative planning-task transformation and then solving the transformed planning task with any planner that determines plans that achieve the goal with maximal probability and always stop in the goal states or "death" or, synonymously, correspond to plans for the original planning task that achieve the goal. It is not a problem for planners to only consider plans with this property. Thus, perhaps surprisingly, planners that do not reason about rewards at all can be used, in conjunction with the multiplicative planning-task transformation, to determine plans that achieve the goal with maximal expected utility. Weaver [3], for example, is an artifi cial intelligence planner based on Bayesian networks, and standard software for Bayesian networks often performs only inferences on probabilities, not rewards. Other artifi cial intelligence planners that reason only with probabilities include [5, 8, 7, 13, 2, 16]. However, the transformed planning task cannot only be solved with planners that determine plans that achieve the goal with maximal probability. After another transformation, it can also be solved with planners that determine plans that achieve the goal with maximal expected reward (as long as they are able to handle rewards that are zero), such as many standard artifi cial intelligence planners including those based an Markov models [4], by declaring "death" another goal state and making the rewards for stopping in goal states other than "death" one and all other rewards zero.

6 Near-Optimal Planning

We have assumed so far that planners are available that determine optimal plans for the transformed planning tasks. However, both the additive and multiplicative planning-task transformation have the following desirable property: the better the plan that planners find for the transformed planning task, the better the corresponding plan is for the original planning task. Thus, both planning-task transformations can be used in conjunction with planners that can determine only near-optimal ("satisficing") plans for the transformed planning tasks. In this section, we analyze the worst-case approximation error of planners that are used in conjunction with the additive and multiplicative planning-task transformation. Remember from Section 4 that the approximation error of a plan that attempts to maximize expected utility is the difference between the certainty equivalent of a plan with maximal expected utility and the certainty equivalent of the plan in question. The worst-case approximation error of a planner is the largest possible difference between the certainty equivalent of a plan with maximal expected utility and the certainty equivalent of the plan found by the planner, over all planning tasks. While the multiplicative planning-task transformation is more general than the additive one, it turns out that the multiplicative planning-task transformation can magnify the worst-case approximation error of a planner whereas the additive one does not. This is a disadvantage of the multiplicative planning-task transformation because, even if a planner has a small worst-case approximation error when used in isolation, its worst-case approximation error can be large when it is used in conjunction with the multiplicative planning-task transformation.

6.1 The Multiplicative Planning-Task Transformation

For the multiplicative planning-task transformation, the probability of not dying during the execution of any plan for the transformed planning task is the same as the expected utility of the corresponding plan for the original planning task. Thus, the larger the probability of not dying during the execution of a plan for the transformed planning task, the larger the expected utility of the corresponding plan for the original planning task, and a near-optimal plan for the transformed planning task corresponds to a near-optimal plan for the original planning task. However, a near-optimal planner with a given worst-case approximation error for the transformed planning task can have a larger worst-case approximation error for the original planning task. Assume, for example, that the optimal plan for the transformed planning task achieves the goal with probability p and stops in only the goal states or "death." Consequently, the expected utility of the optimal plan that achieves the goal for the original planning task is p and its certainty equivalent is $\log_\gamma p$. Case 1: Consider a near-optimal planner with absolute (= additive) approximation error for the transformed planning task. This means that the planner determines a plan whose quality is at least $x - \epsilon$ for a constant $\epsilon > 0$ if the quality of the best plan is x. If this planner is used to solve the transformed planning task, it can potentially determine a plan that stops in only the goal states or "death" and whose probability of goal achievement is only $p - \epsilon$ (but not worse). Thus, the corresponding plan for the original planning task achieves the goal but its expected utility is only $p - \epsilon$ and its certainty equivalent is only $\log_\gamma[p - \epsilon]$. To determine the approximation error

of this plan for the original planning task, we need to consider how close its certainty equivalent is to the certainty equivalent of the plan that achieves the goal with maximal expected utility. Table 1 summarizes this data.

Table 1. Approximation Error for Case 1

	probability of goal achievement for the transformed planning task	certainty equivalent for the original planning task
optimal plan	p	$\log_\gamma p$
found plan (worst case)	$p - \epsilon$	$\log_\gamma [p - \epsilon]$

Thus, the resulting approximation error for the original planning task, $\log_\gamma p - \log_\gamma [p - \epsilon] = \log_\gamma \frac{p}{p-\epsilon}$, increases as p and γ decrease. It can get arbitrarily large. Case 2: Now consider a near-optimal planner with relative (= multiplicative) approximation error for the transformed planning task. This means that the planner determines a plan whose quality is at least $(1 - \epsilon)x$ for a constant $\epsilon > 0$ if the quality of the best plan is x. Table 2 summarizes the resulting data.

Table 2. Approximation Error for Case 2

	probability of goal achievement for the transformed planning task	certainty equivalent for the original planning task
optimal plan	p	$\log_\gamma p$
found plan (worst case)	$(1 - \epsilon)p$	$\log_\gamma [1 - \epsilon] + \log_\gamma p$

Thus, the resulting approximation error for the original planning task, $\log_\gamma p - \log_\gamma [1 - \epsilon] - \log_\gamma p = \log_\gamma \frac{1}{1-\epsilon}$, is additive. Figure 6 shows its graph.

6.2 The Additive Planning-Task Transformation

For the additive planning-task transformation, the reward of any sequential plan for the transformed planning task is the same as the certainty equivalent of the corresponding plan for the original planning task. Thus, the larger the reward of a plan for the transformed planning task, the larger the certainty equivalent of the corresponding plan for the original planning task, and a near-optimal plan for the transformed planning task corresponds to a near-optimal for the original planning task. Furthermore, a near-optimal planner that has a given absolute or relative worst-case approximation error for the transformed planning task has the same absolute or relative worst-case approximation error, respectively, for the original planning task.

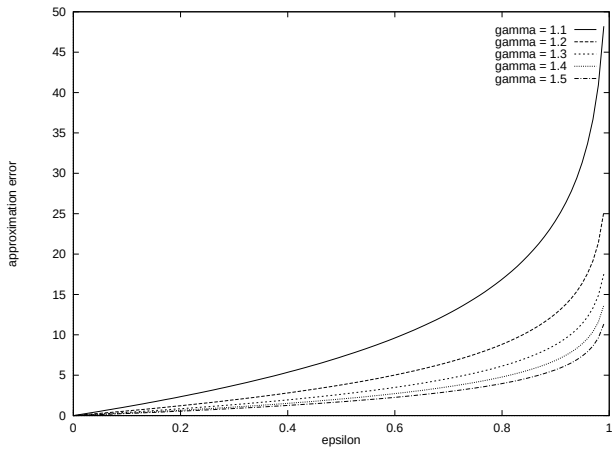

Fig. 6. Approximation Error for Case 2

7 Conclusions

Many existing artificial intelligence planners attempt to determine plans that achieve the goal with maximal probability or minimal expected execution time, but agents often need to determine plans that achieve the goal with maximal expected utility for nonlinear utility functions. In this paper, we developed a planning methodology for determining plans that achieve the goal with maximal expected utility for convex exponential utility functions. These utility functions are necessary to model the immediate soft deadlines often encountered in the context of delivery tasks. Our planning methodology combines constructive approaches from artificial intelligence with more descriptive approaches from utility theory. It is based on simple representation changes, that we call the additive and multiplicative planning-task transformations. Advantages of the planning-task transformations are that they are fast, scale well, allow for optimal and near-optimal planning, and are grounded in utility theory. Future work includes studying planning for agents with even more general preference models, including trading-off between different resource consumptions (such as money, energy, and time).

8 Acknowledgments

The Intelligent Decision-Making Group is supported by an NSF Career Award to Sven Koenig under contract IIS-9984827. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations and agencies or the U.S. Government.

References

1. D. Bernoulli. Specimen theoriae novae de mensura sortis. *Commentarii Academiae Scientiarum Imperialis Petropolitanae*, 5, 1738. Translated by L. Sommer, *Econometrica*, 22: 23–36, 1954.
2. A. Blum and J. Langford. Probabilistic planning in the graphplan framework. In *Proceedings of the European Conference on Planning*, 1999.
3. J. Blythe. Planning with external events. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 94–101, 1994.
4. C. Boutilier, T. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94, 1999.
5. J. Bresina and M. Drummond. Anytime synthetic projection: Maximizing the probability of goal satisfaction. In *Proceedings of the National Conference on Artificial Intelligence*, pages 138–144, 1990.
6. T. Dean, J. Firby, and D. Miller. Hierarchical planning involving deadlines, travel times, and resources. *Computational Intelligence*, 4(4):381–398, 1988.
7. D. Draper, S. Hanks, and D. Weld. Probabilistic planning with information gathering. In *Proceedings of the International Conference on Artificial Intelligence Planning Systems*, pages 31–37, 1994.
8. R. Goldman and M. Boddy. Epsilon-safe planning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 253–261, 1994.
9. P. Haddawy and S. Hanks. Representation for decision-theoretic planning: Utility functions for deadline goals. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, 1992.
10. R. Howard and J. Matheson. Risk-sensitive Markov decision processes. *Management Science*, 18(7):356–369, 1972.
11. K. Kanazawa and T. Dean. A model for projection and action. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 985–990, 1989.
12. S. Koenig and R.G. Simmons. How to make reactive planners risk-sensitive. In *Proceedings of the International Conference on Artificial Intelligence Planning Systems*, pages 293–298, 1994.
13. N. Kushmerick, S. Hanks, and D. Weld. An algorithm for probabilistic planning. *Artificial Intelligence*, 76(1–2):239–286, 1995.
14. S. Marcus, E. Fernandez-Gaucherand, D. Hernandez-Hernandez, S. Colaruppi, and P. Fard. Risk-sensitive Markov decision processes. In C. Byrnes et. al., editor, *Systems and Control in the Twenty-First Century*, pages 263–279. Birkhauser, 1997.
15. R. Neuneier and O. Mihatsch. Risk sensitive reinforcement learning. In *Proceedings of the Neural Information Processing Systems*, pages 1031–1037, 1999.
16. N. Onder and M. Pollack. Conditional, probabilistic planning: A unifying algorithm and effective search control machanisms. In *Proceedings of the National Conference on Artificial Intelligence*, 1999.
17. J. Pratt. Risk aversion in the small and in the large. *Econometrica*, 32(1-2):122–136, 1964.
18. J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, second edition, 1947.
19. M. Wellman, M. Ford, and K. Larson. Path planning under time-dependent uncertainty. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 523–539, 1995.
20. P. Whittle. *Risk-Sensitive Optimal Control*. Wiley, 1990.