Appears in the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK), 1998

Evaluating Options in a Context

John F. Horty*
Philosophy Department and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742
horty@umiacs.umd.edu

Martha E. Pollack†
Department of Computer Science
and Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15260
pollack@cs.pitt.edu

1 Introduction

The theory of rational choice, as formulated in the economic and philosophical literature, assumes that agents evaluate alternative actions by reference to a probability distribution over their possible outcomes together with a utility function defined on those outcomes: in the simplest case, the agent combines probability and utility into a notion of expected utility defined over actions, and then chooses some action whose expected utility is maximal. A good deal of attention has been devoted to the structure of those utility functions that might actually be thought to underlie human decision making [9], and the more applied literature on decision analysis has focused on the task of eliciting such preference information from humans [8]. When we attempt to build artificial agents that are capable of making rational decisions, we likewise need to provide them with techniques for evaluating the options they encounter.

Our approach to this problem—the rational evaluation of options—differs in two important ways from that of classical decision theory. First, while decision theory assumes that the utility of an outcome is given as part of the background setting, we note that the overall desirability of an option presented to an agent is often not immediately apparent; and we are explicitly concerned with the mechanism through which it might be discovered. We focus, in particular, on the case in which the option presented to an agent has a known benefit, but requires some effort—the execution of a plan—for its achievement. In order to evaluate the overall desirability of the option, the agent thus has to arrive at some assessment of the cost involved in achieving it.

Second, we insist that the task of evaluating an option should be computationally realizable; and in particular, our work here is developed within the theoretical framework first articulated in [2], and then further elaborated in [3, 16], according to which it is best to view a resource-bounded agent as operating always against the background of some current set of intentions, or plans. In contrast to standard decision theory, where actions are evaluated in isolation, we develop a model in which the options presented to an agent are evaluated against a background context provided by the agent's current plans—commitments to future activities, which, at any given point, may themselves be only partially specified. The interactions

*Supported by National Science Foundation grant IRI-9619562.
†Supported by National Science Foundation grants IRI-9258392 and IRI-9619579, and by the Air Force Office of Scientific Research Contract F49620-98-1-0436.

between the new option and the background context can complicate the task of evaluating the option, rendering it either more or less desirable in context than it would have been in isolation.

As an example, suppose an agent is already committed to going to the airport tomorrow afternoon to catch a plane, but has not yet decided whether to get there by taxi or by taking the airport shuttle van. Given this background context, the agent might then have to evaluate the newly presented option of attending a lunch time meeting tomorrow. If the meeting is to be held on campus, and is likely to run late, a decision to attend may rule out the possibility of taking the van. Assuming that the van costs less than the taxi, the new option would then be less desirable in context that it would have been in isolation; the benefit of attending the meeting must be at least great enough to compensate for the difference in cost between taxi and van to make it worthwhile. On the other hand, suppose the meeting is to be held at an airport hotel. In this case, the background context reduces the cost associated with the new option, increasing its overall desirability, since the agent is already committed to going to the airport: the agent might rationally choose to attend the meeting, since he is going to the airport anyway, even if this option is not one the agent would have decided to pursue in isolation.

The present paper begins the task of providing a theoretical and computational analysis of the reasoning involved in situations like this, where a new option must be evaluated within the context provided by a background plan. We believe that the approach developed here has some applicability in the analysis of human choice. However, because we are primarily concerned with the design of artificial agents, we represent both the agent's background context and the new options it might encounter using a well-understood plan formalism familiar from artificial intelligence (AI). Our approach to evaluating the desirability of new plans can thus be dovetailed with computational accounts of plan generation.

We limit our attention in this paper to a very restricted setting, in which all plans are primitive (not hierarchical) and complete, and all actions have deterministic outcomes. In this simple setting, the only ways in which one plan can influence the cost of another is by allowing or blocking the possibility that separate steps might be merged into one. (In the airport story, for instance, when the meeting is held at the airport, the step of getting to the meeting can be merged with the step of getting to the airport, which is already part of the agent's background plan.) Although our restriction to this special case prevents us from considering many of the more interesting ways in which plans might interact, even this very simple setting is sufficiently rich to allow us to illustrate the shape of our theory, and we defer a detailed treatment of more complicated plan interactions to subsequent work.

2 Primitive plans

Basic concepts

We represent primitive plans using a standard formalism [13, 20, 15], in which a plan consist of a set of steps, temporal constraints on those steps, and causal links, which record dependency relations among steps. As usual, we assume a set of action types, defined in terms of preconditions and effects (for clarity, we limit our attention only to propositional preconditions and effects). The plan steps are instances of the action types. The planning literature tends to concentrate on qualitative temporal constraints, specifying only the relative order of steps (but see [1]). In contrast, we also allow for quantitative constraints, which associate steps with actual time points. To this end, we model time as a totally ordered set of moments $\{m_0, m_1, \ldots\}$, where $m_i < m_j$ if and only if $i < j$, and we assume here that each step occupies a single moment of time.

Definition 1 (Primitive plan) A *primitive plan* $\mathcal{P}$ is a triple of the form $\langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$, with these components defined as follows: $\mathcal{S}$ is a set of steps of the form S_i, each associated with a time indicator t_i; $\mathcal{O}$ is a set of ordering constraints, of the form $t_i = t_j$, $t_i < t_j$, $t_i = m_k$, or $t_i < m_k$, where t_i and t_j are time indicators associated with steps belonging to $\mathcal{S}$ and m_k is a moment; $\mathcal{L}$ is a set of causal links of the form $\langle S_i, Q, S_j \rangle$, where Q is an effect of the step S_i and a precondition of the step S_j.

We assume a function *Type* associating each step S_i with $Type(S_i)$, its action type. We require $\mathcal{O}$ to contain a temporal constraint of the form $t_i < t_j$ whenever there is a link $\langle S_i, Q, S_j \rangle$ in $\mathcal{L}$. And we suppose that an entailment relation $\vdash$ is defined on the temporal constraint language, allowing us to draw out implicit consequences (for example, $\{t_j = m, t_i < t_j\} \vdash t_i < m$), and providing us, also, with a notion of consistency for a set of temporal constraints.

To illustrate, let us consider the plan of buying a shirt at the mall, which might be represented in the current framework as $\mathcal{P}_1 = \langle \mathcal{S}_1, \mathcal{O}_1, \mathcal{L}_1 \rangle$, where $\mathcal{S}_1 = \{S_1, S_2, S_3, S_4\}$, $\mathcal{O}_1 = \{t_1 < t_3, t_2 < t_3, t_3 < t_4, t_3 = m_6\}$, and $\mathcal{L}_1 = \{\langle S_1, A, S_3 \rangle, \langle S_2, B, S_3 \rangle, \langle S_3, C, S_4 \rangle\}$. Step S_1 represents an action of going to the mall and that S_2 is an action of bringing one's wallet (which, we suppose, includes a credit card); S_1 thus has as its effect the statement A, representing the proposition that the agent is at the mall, and S_2 the statement B, representing the proposition that the agent has its credit card. The step S_3, then, represents the action of actually buying the shirt; it requires A and B as preconditions and generates C as an effect, representing the proposition that the agent has the shirt. Finally, S_4 is a dummy step representing the achievement of the goal, taking C as its precondition and generating no effects. The links $\langle S_1, A, S_3 \rangle$ and $\langle S_2, B, S_3 \rangle$ represent the fact that S_1 and S_2 are performed for the purpose of establishing the various preconditions of S_3 (in the process of plan generation, their presence would block the insertion of another step that might interfere with these preconditions), and likewise the link $\langle S_3, C, S_4 \rangle$ indicates that S_3 is performed in order to achieve the goal state. Of the temporal ordering constraints, the first three are qualitative constraints inherited from the causal links, while the fourth is a quantitative constraint specifying that S_3 must be performed precisely at m_6 (the moment, perhaps, that the shirt is on sale).

We will say that a plan is scheduled when each of its steps has been assigned a specific moment of execution. In this paper, we prohibit schedules with concurrent actions, although, importantly, two steps of the same action type can be *merged*—assigned to the same moment of execution.

Definition 2 (Schedule, scheduled and schedulable plans) A *schedule* for a plan $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ is a set of constraints $\overline{\mathcal{O}}$ such that: (1) there is a constraint of the form $t_i = m$ in $\overline{\mathcal{O}}$ for each S_i in $\mathcal{S}$; (2) $\mathcal{O} \cup \overline{\mathcal{O}}$ is consistent; and (3) $Type(S_i) = Type(S_j)$ whenever $\mathcal{O} \cup \overline{\mathcal{O}} \vdash t_i = t_j$. The plan $\mathcal{P}$ is said to be *scheduled* whenever there exists a set of constraints $\overline{\mathcal{O}} \subseteq \mathcal{O}$ such that $\overline{\mathcal{O}}$ is a schedule for the plan; $\mathcal{P}$ is said to be *schedulable* whenever there exists a schedule for it.

As an example, the constraint set $\overline{\mathcal{O}} = \{t_1 = m_3, t_2 = m_4, t_3 = m_6, t_4 = m_7\}$ is a schedule for the plan $\mathcal{P}_1$ above, showing that this plan is schedulable. Of course, a plan whose ordering constraints are themselves inconsistent cannot be scheduled, but even a plan whose ordering constraints are consistent may nevertheless fail to be schedulable, since its only consistent linearizations may be those in which type distinct steps are assigned to the same moment. Schedulability is thus a stronger requirement than mere consistency of temporal constraints.

We focus in this paper on plans that are complete, in the sense that no further planning is needed in order to guarantee the preconditions of their various steps, although additional scheduling may still be required.

Definition 3 (Complete plans) Let $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ be a plan. A precondition A of a step S_i from $\mathcal{S}$ is *established* whenever there is some link $\langle S_j, A, S_i \rangle$ in $\mathcal{L}$. A link $\langle S_j, A, S_i \rangle$ from $\mathcal{L}$ is *threatened* whenever

there is both an action S_k in $\mathcal{S}$ with effect $\neg A$ and a schedule $\overline{\mathcal{O}}$ for $\mathcal{P}$ such that $\mathcal{O} \cup \overline{\mathcal{O}} \vdash t_j < t_k < t_i$. The plan $\mathcal{P}$ is *complete* just in case each precondition of each step from $\mathcal{S}$ is established and no link from $\mathcal{L}$ is threatened.

This definition of plan completeness is equivalent to the standard notion from the literature, except that it replaces the idea of temporal consistency with the stronger notion of schedulability.

In order to assess the desirability of a new option against a background context, we need to be able to reason about the plans that are formed when two others are combined, as follows.

Definition 4 (Union of plans) Given plans $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ and $\mathcal{P}' = \langle \mathcal{S}', \mathcal{O}', \mathcal{L}' \rangle$, the *union* of the two plans is $\mathcal{P} \cup \mathcal{P}' = \langle \mathcal{S} \cup \mathcal{S}', \mathcal{O} \cup \mathcal{O}', \mathcal{L} \cup \mathcal{L}' \rangle$.

Note that the union of two independently schedulable plans might not be schedulable, since their temporal constraint sets may not even be jointly consistent; also, the union of two complete plans might not be complete, since steps in one may threaten links in the other. If the union of two complete plans can be made complete and schedulable simply through the addition of ordering constraints, we say that the plans are strongly compatible.

Definition 5 (Strong compatibility) Let $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ and $\mathcal{P}' = \langle \mathcal{S}', \mathcal{O}', \mathcal{L}' \rangle$ be complete plans. Then $\mathcal{P}$ and $\mathcal{P}'$ are *strongly compatible* just in case there is a temporal constraint set $\mathcal{O}''$ such that $\langle \mathcal{S} \cup \mathcal{S}', \mathcal{O} \cup \mathcal{O}' \cup \mathcal{O}'', \mathcal{L} \cup \mathcal{L}' \rangle$ is complete and schedulable.

As an example, consider the plan $\mathcal{P}_2 = \langle \mathcal{S}_2, \mathcal{O}_2, \mathcal{L}_2 \rangle$, where $\mathcal{S}_2 = \{S_5, S_6\}$, $\mathcal{O}_2 = \{t_5 < t_6\}$, and $\mathcal{L}_2 = \{\langle S_5, D, S_6 \rangle\}$, and where the step S_5 has D and $\neg A$ as effects. (Intuitively, $\mathcal{P}_2$ might represent the plan of going home, with D representing the proposition that the agent is at home, and $\neg A$, of course, the proposition that the agent is no longer at the mall.) Then $\mathcal{P}_2$ and the previous $\mathcal{P}_1$ are strongly compatible, as shown by the constraint set $\mathcal{O}'' = \{t_3 < t_5\}$, since $\langle \mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{O}'', \mathcal{L}_1 \cup \mathcal{L}_2 \rangle$ is complete and schedulable.

The notion of strong compatibility defined here is, in fact, a very strong notion, since it does not allow either of two compatible plans to be modified in any way, but only supplemented with additional scheduling information, in order for their joint execution to be guaranteed. This notion is not, however, the strongest available. A stronger notion is that of perfect compatibility, where two complete plans $\mathcal{P}$ and $\mathcal{P}'$ are defined as *perfectly compatible* just in case their union $\mathcal{P} \cup \mathcal{P}'$ is itself complete and schedulable. It is easy to see that $\mathcal{P}_1$ and $\mathcal{P}_2$, though strongly compatible, are not perfectly compatible, since the joint plan $\mathcal{P}_1 \cup \mathcal{P}_2$ allows for schedules in which S_5 occurs between S_1 and S_3, threatening the link $\langle S_1, A, S_3 \rangle$.

Semantics

Eventually, we will want to interpret a plan as specifying a set of allowed futures—intuitively, those futures consistent with an execution of the plan. (An interpretation along these lines can be developed within the general logical framework of branching time [18, 19].) For reasons of space, however, we restrict ourselves in this paper to a simpler account, in which complete and scheduled plans, rather than futures, are taken as the points in the semantic space, and more abstract plans are associated with sets of these.

We begin by adapting the notion of refinement [10] from the plan generation literature.

Definition 6 (Refinement; $\sqsubseteq$) Let $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ and $\mathcal{P}' = \langle \mathcal{S}', \mathcal{O}', \mathcal{L}' \rangle$ be plans. Then $\mathcal{P}'$ is a *refinement* of $\mathcal{P}$ ($\mathcal{P} \sqsubseteq \mathcal{P}'$) just in case $\mathcal{S} \subseteq \mathcal{S}'$ and $\mathcal{O} \subseteq \mathcal{O}'$ and $\mathcal{L} \subseteq \mathcal{L}'$.

Letting Π represent the set of complete and scheduled plans, we define the semantic interpretation of a plan as follows.

Definition 7 (Interpretation; $v[\mathcal{P}]$) The *interpretation* of a plan $\mathcal{P}$ is the set of its complete and scheduled refinements: $v[\mathcal{P}] = \{\mathcal{P}' : \mathcal{P} \sqsubseteq \mathcal{P}'\} \cap \Pi$.

The idea, of course, is that a plan is to be interpreted as the set of ways in which it might be carried out, and so it is natural to define a plan as consistent whenever there is some way in which it can be carried out.

Definition 8 (Plan consistency) A plan $\mathcal{P}$ is *consistent* just in case $v[\mathcal{P}] \neq \emptyset$.

Note that a complete plan is consistent just in case it is schedulable, and that an incomplete plan is consistent just in case it has a complete and schedulable refinement.

3 Evaluation of options

For the purposes of this paper, we define an *option* as a complete plan that is presented to an agent for acceptance or rejection. This terminology may seem peculiar, since it is often natural to think of an option as something more along the lines of a goal state—having a new shirt, say. Nevertheless, even when the value of an option lies entirely in the achievement of some goal state, sensible reasoning demands that goal states and their means of achievement—in this case, a trip to the mall—must be evaluated together.

We suppose that an agent evaluates each new option $\mathcal{P}$ against the background of a context $\mathcal{C}$, some plan to which it is already committed, and that the process of evaluation proceeds as follows. First, the agent determines whether $\mathcal{P}$ is compatible with $\mathcal{C}$—where, for the purposes of this paper, we will assume that the concept of compatibility can be usefully approximated through our notion of strong compatibility—and if not, $\mathcal{P}$ is rejected. Of course, this policy of immediately rejecting incompatible options is a considerable simplification. More realistically, an agent faced with an incompatible option $\mathcal{P}$ could explore either local revisions to the plan that might guarantee compatibility, or else alternative plans for achieving the goal that $\mathcal{P}$ aims at; and if the goal is valuable, the agent might also consider modifications of his background context. However, we cannot examine these more sophisticated alternatives in the present paper.

Assuming compatibility, then, the agent should accept the new option just in case its benefit outweighs its cost in the context. Again, we simplify by supposing that the benefit of the option $\mathcal{P}$—represented here as $\beta(\mathcal{P})$—is both apparent and independent of context (in the most natural case, this benefit will derive from the goal state at which the plan is directed). All that remains to be specified, then, is the cost of the new option $\mathcal{P}$ in the context $\mathcal{C}$.

Cost in isolation

We begin by defining the cost of a plan in isolation. We take as given a function *Cost* mapping action types into real numbers representing their costs, and assume that the function is extended to the steps of a plan in the natural way: $Cost(S_i) = Cost(Type(S_i))$.

Next, we introduce an auxiliary notion of point cost, defined only for complete, scheduled plans—the points in the semantic space. Where $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ is such a plan, we partition the plan steps into sets of actions forced (by the temporal constraints) to occur at the same moment, taking $[S_i] = \{S_j : \mathcal{O} \vdash t_i = t_j\}$ for each $S_i \in \mathcal{S}$. We then let $[\mathcal{P}]$ represent the set of these equivalence classes: $[\mathcal{P}] = \{[S_i] : S_i \in \mathcal{S}\}$. It follows from our definition of a schedule that steps in the same equivalence class will necessarily represent

actions of the same type; these type-identical steps performed at the same moment are to be thought of as collapsing into a single merged step. We therefore define the point cost of the plan itself as the sum of the costs assigned to the merged steps it contains:

$$\textit{Point-cost}(\mathcal{P}) = \sum_{[S_i] \in [\mathcal{P}]} \textit{Cost}(S_i).$$

Given this auxiliary notion, it is now natural to define the cost of an arbitrary consistent plan as the point cost of the least expensive way in which it might be carried out, that is, the least expensive point in its semantic interpretation.

Definition 9 (Cost of a plan; $\kappa(\mathcal{P})$) Where $\mathcal{P}$ is a consistent plan, the *cost* of $\mathcal{P}$ is the point cost of its least expensive complete and scheduled refinement: $\kappa(\mathcal{P}) = \min\{\textit{Point-cost}(\mathcal{P}') : \mathcal{P}' \in v[\mathcal{P}]\}$.

It is easy to see that $\kappa(\mathcal{P}) = \textit{Point-cost}(\mathcal{P})$ whenever $\mathcal{P}$ is itself a complete and scheduled plan, and that $\kappa(\mathcal{P}_\emptyset) = 0$ for the null plan $\mathcal{P}_\emptyset = \langle \emptyset, \emptyset, \emptyset \rangle$.

Cost in context

Having defined the cost of a plan in isolation, we now turn to our central task of defining the cost of a new option $\mathcal{P}$ in the context of a background plan $\mathcal{C}$. Our treatment of this concept is simple: we take the cost of the new option in context to be its *marginal* cost—the cost of carrying out $\mathcal{P}$ along with $\mathcal{C}$, less the cost of carrying out $\mathcal{C}$ alone.

Definition 10 (Cost of a plan in a context; $\kappa(\mathcal{P}/\mathcal{C})$) Where the plans $\mathcal{C}$ and $\mathcal{P}$ are strongly compatible, the *cost of $\mathcal{P}$ in the context $\mathcal{C}$* is $\kappa(\mathcal{P}/\mathcal{C}) = \kappa(\mathcal{P} \cup \mathcal{C}) - \kappa(\mathcal{C})$.

It follows immediately from this definition that the cost of a plan in the null context is identical to its cost in isolation: $\kappa(\mathcal{P}/\mathcal{P}_\emptyset) = \kappa(\mathcal{P})$. It is also worth noting that the cost of a plan in any context that already includes that plan as a component is zero: $\kappa(\mathcal{P}/\mathcal{P} \cup \mathcal{C}) = 0$.

This definition can be illustrated with a case in which the cost of a new option is actually affected by the background context. Suppose the agent's background context is simply the plan to buy a shirt at the mall, represented by our earlier $\mathcal{P}_1$, and imagine that the agent is presented with the new option of going to the mall for some swim goggles. More exactly, we can take the new option as the plan $\mathcal{P}_3 = \langle \mathcal{S}_3, \mathcal{O}_3, \mathcal{L}_3 \rangle$, where $\mathcal{S}_3 = \{S_7, S_8, S_9, S_{10}\}$, $\mathcal{O}_3 = \{t_7 < t_9, t_8 < t_9, t_9 < t_{10}\}$, and $\mathcal{L}_3 = \{\langle S_7, A, S_9 \rangle, \langle S_8, B, S_9 \rangle, \langle S_9, E, S_{10} \rangle\}$. Here, the steps S_7 and S_8 again represent actions of going to the mall and bringing one's wallet, steps sharing the respective types of S_1 and S_2 from the background plan $\mathcal{P}_1$; the step S_9 represents the action of purchasing the goggles; S_{10} is again a dummy step representing goal achievement; and the statement E represents the proposition that the agent has swim goggles. Let us suppose that these various steps carry the following costs: each of S_2, S_3, S_8, and S_9 carries a cost of 1, since both carrying a wallet and making a purchase are easy to do; each of S_1 and S_7 carries a cost of 10, since any trip to the mall is abhorrent; and S_4 and S_{10}, as dummy steps, both carry a cost of 0.

Given this information, it is clear that $\kappa(\mathcal{P}_1) = 12$—the cost of the agent's background plan is 12. Presumably, then, the benefit of this background plan must be at least 12—we must have $\beta(\mathcal{P}_1) \geq 12$—or the agent would not have adopted it. Suppose, however, that $\beta(\mathcal{P}_3) = 2$. It is clear also that $\kappa(\mathcal{P}_3) = 12$, so that, considered in isolation, the new option would not be worth pursuing. On the other hand, it is easy to see that $\kappa(\mathcal{P}_3 \cup \mathcal{P}_1) = 13$, since the least expensive execution of the joint plan, in which both the steps S_1 and S_7 as well as the steps S_2 and S_8 are merged, carries a cost of 13. Therefore, we have

$\kappa(\mathcal{P}_3/\mathcal{P}_1) = \kappa(\mathcal{P}_3 \cup \mathcal{P}_1) - \kappa(\mathcal{P}_1) = 1$. Even though the new option would not be worth pursuing in isolation, it is worth pursuing in context, since its benefit is greater than its cost in context.

As this example shows, the cost of a plan in context may be less than its cost in isolation, but it is also possible for the cost is context to be greater. Our earlier taxi/van story already illustrates this possibility, but it is worth noting that it also arises even in the more restricted framework of complete, strongly compatible plans. In this setting, a plan in context will have a higher cost than it has in isolation if the background plan contains steps that might be merged if it were performed in isolation, but the new option blocks that merge possibility. For instance, suppose the agent already intends to purchase a shirt and some swim goggles at the mall, and is now considering an option of seeing a movie. Suppose further that the movie begins soon, leaving time to make only one of the purchases, and that the mall stores are closed after the movie lets out. Then attending the movie, in this context, has an extra cost, since it means an additional trip to the mall to carry out the already intended plans. Just as in the taxi/van story, the present example illustrates a case in which a new option is more expensive in context than in isolation because its adoption would rule out the most efficient executions of the background plan.

Cost estimates

Although the notion of cost as the least expensive method of execution is defined for any consistent plan, we do not necessarily assume that the agent knows the true cost either of his background plan or of any new options under consideration. Instead, the agent may only estimate the cost of its plans.

Definition 11 (Cost estimate for a plan) Where $\mathcal{P}$ is a consistent plan, a *cost estimate* for $\mathcal{P}$ is an interval of the form $\epsilon = [\epsilon^-, \epsilon^+]$, where ϵ^- and ϵ^+ are nonnegative real numbers such that $\epsilon^- \leq \kappa(\mathcal{P}) \leq \epsilon^+$.

Cost estimates, so defined, accurately bound the actual cost of a plan, and are thus related to the interval measures of plan cost used in the decision-theoretic plan generation literature [21, 7, 6].

We now show that, under certain coherence conditions, a cost estimate for a plan in context can be derived from a cost estimate for the context together with a cost estimate for the plan and context combined. Assume that $\mathcal{P}$ and $\mathcal{C}$ are strongly compatible plans, and that $\epsilon_\mathcal{C} = [\epsilon^-_\mathcal{C}, \epsilon^+_\mathcal{C}]$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}} = [\epsilon^-_{\mathcal{P}\cup\mathcal{C}}, \epsilon^+_{\mathcal{P}\cup\mathcal{C}}]$ are cost estimates for the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$ respectively. We know from the definition of a cost estimate that $\epsilon^-_\mathcal{C} \leq \epsilon^+_\mathcal{C}$ and $\epsilon^-_{\mathcal{P}\cup\mathcal{C}} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}}$, but the definition tells us nothing about the relations among the intervals $\epsilon_\mathcal{C}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ themselves. Nevertheless, it is reasonable to conclude that $\epsilon^-_\mathcal{C} \leq \epsilon^-_{\mathcal{P}\cup\mathcal{C}}$, since the least expensive execution of the compound plan $\mathcal{P} \cup \mathcal{C}$ cannot be less costly than the least expensive execution of $\mathcal{C}$, one of its components; and similarly, $\epsilon^+_\mathcal{C} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}}$. We characterize the pair of estimates $\epsilon_\mathcal{C}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ as *jointly coherent* just in case these two conditions hold: $\epsilon^-_\mathcal{C} \leq \epsilon^-_{\mathcal{P}\cup\mathcal{C}}$ and $\epsilon^+_\mathcal{C} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}}$.

As long as $\epsilon_\mathcal{C}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ are jointly coherent we can derive a cost estimate $\epsilon_{\mathcal{P}/\mathcal{C}} = [\epsilon^-_{\mathcal{P}/\mathcal{C}}, \epsilon^+_{\mathcal{P}/\mathcal{C}}]$ for the plan $\mathcal{P}$ in the context $\mathcal{C}$ in the following way. Given joint coherence, the end points of the intervals $\epsilon_\mathcal{C}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ can stand in only two possible ordering relations:

$$(1) \quad \epsilon^-_\mathcal{C} \leq \epsilon^+_\mathcal{C} \leq \epsilon^-_{\mathcal{P}\cup\mathcal{C}} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}},$$
$$(2) \quad \epsilon^-_\mathcal{C} \leq \epsilon^-_{\mathcal{P}\cup\mathcal{C}} \leq \epsilon^+_\mathcal{C} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}}.$$

In either case, it is clear that $\epsilon^+_{\mathcal{P}/\mathcal{C}}$ should be defined as $\epsilon^+_{\mathcal{P}\cup\mathcal{C}} - \epsilon^-_\mathcal{C}$, the maximum possible distance between points in $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ and $\epsilon_\mathcal{C}$. In case (1), we know that $\epsilon^-_{\mathcal{P}/\mathcal{C}}$ should likewise defined as $\epsilon^-_{\mathcal{P}\cup\mathcal{C}} - \epsilon^+_\mathcal{C}$, the minimum possible distance. In case (2), it is reasonable to take $\epsilon^-_{\mathcal{P}/\mathcal{C}}$ as 0, since we know, even when the low estimate for executing $\mathcal{P} \cup \mathcal{C}$ is less than the high estimate for executing $\mathcal{C}$, that the true cost of executing $\mathcal{P} \cup \mathcal{C}$ can

be no less than the true cost of executing $\mathcal{C}$. Combining cases (1) and (2), we can therefore take $\epsilon^-_{\mathcal{P}/\mathcal{C}}$ as $\max[0, \epsilon^-_{\mathcal{P}\cup\mathcal{C}} - \epsilon^+_{\mathcal{C}}]$, leading to the following general definition.

Definition 12 (Cost estimate for a plan in context) Where the plans $\mathcal{P}$ and $\mathcal{C}$ are strongly compatible, let $\epsilon_{\mathcal{C}} = [\epsilon^-_{\mathcal{C}}, \epsilon^+_{\mathcal{C}}]$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}} = [\epsilon^-_{\mathcal{P}\cup\mathcal{C}}, \epsilon^+_{\mathcal{P}\cup\mathcal{C}}]$ be a pair of jointly coherent cost estimates for the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$. Then the *cost estimate for the plan $\mathcal{P}$ in the context $\mathcal{C}$* is the interval $\epsilon_{\mathcal{P}/\mathcal{C}} = [\epsilon^-_{\mathcal{P}/\mathcal{C}}, \epsilon^+_{\mathcal{P}/\mathcal{C}}]$, where $\epsilon^-_{\mathcal{P}/\mathcal{C}} = \max[0, \epsilon^-_{\mathcal{P}\cup\mathcal{C}} - \epsilon^+_{\mathcal{C}}]$ and $\epsilon^+_{\mathcal{P}/\mathcal{C}} = \epsilon^+_{\mathcal{P}\cup\mathcal{C}} - \epsilon^-_{\mathcal{C}}$.

It follows immediately from this definition that $\kappa(\mathcal{P}/\mathcal{C})$, the true cost of $\mathcal{P}$ in the context $\mathcal{C}$, lies within the derived interval $\epsilon_{\mathcal{P}/\mathcal{C}}$; and it is also easy to see that the derived interval $\epsilon_{\mathcal{P}/\mathcal{C}}$ narrows monotonically as the intervals $\epsilon_{\mathcal{C}}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ are narrowed.

The derived interval estimate of cost in context is useful because, in many cases, it allows an agent to accept or reject an option without calculating its true cost. Suppose, for example, that an agent with background plan $\mathcal{C}$ is considering the new option $\mathcal{P}$ with benefit $\beta(\mathcal{P})$; and imagine that the agent has assigned estimated costs $\epsilon_{\mathcal{C}}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ to the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$, from which it derives the estimate $\epsilon_{\mathcal{P}/\mathcal{C}} = [\epsilon^-_{\mathcal{P}/\mathcal{C}}, \epsilon^+_{\mathcal{P}/\mathcal{C}}]$ for the cost of $\mathcal{P}$ in the context $\mathcal{C}$. Then if $\beta(\mathcal{P}) > \epsilon^+_{\mathcal{P}/\mathcal{C}}$, the agent is justified in adopting the new option, since the cost in context of the option is necessarily less than its benefit; and likewise, the agent is justified in rejecting the option if $\beta(\mathcal{P}) < \epsilon^-_{\mathcal{P}/\mathcal{C}}$, since its cost in context is necessarily greater than its benefit. If $\epsilon^-_{\mathcal{P}/\mathcal{C}} \leq \beta(\mathcal{P}) \leq \epsilon^+_{\mathcal{P}/\mathcal{C}}$, there are two subcases to consider. First, if it happens that $\epsilon^-_{\mathcal{P}/\mathcal{C}} = \epsilon^+_{\mathcal{P}/\mathcal{C}}$, then, since we know that $\kappa(\mathcal{P}/\mathcal{C})$ lies within the interval $\epsilon_{\mathcal{P}/\mathcal{C}}$, it follows that $\beta(\mathcal{P}) = \kappa(\mathcal{P}/\mathcal{C})$, and so the agent is justified either in accepting or rejecting the option. If $\epsilon^-_{\mathcal{P}/\mathcal{C}} < \epsilon^+_{\mathcal{P}/\mathcal{C}}$, on the other hand, the agent's interval estimates do not provide enough information to determine whether the option should be adopted or rejected. In this last case, and only this case, the agent is forced to refine his estimates further before making a rational decision, narrowing his cost estimates for $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$, and thereby also narrowing his derived estimate for $\mathcal{P}$ in the context of $\mathcal{C}$.

4 Reasoning procedures

We now present some algorithms through which the process sketched in Section 3 of evaluating a new option $\mathcal{P}$ against the background of a context $\mathcal{C}$ might actually be accomplished.

As explained earlier, the first step of the process is determining whether $\mathcal{P}$ is, in fact, strongly compatible with the context $\mathcal{C}$. This problem has been studied in [22], which develops a constraint satisfaction procedure to determine plan compatibility. This algorithm, COMBINE, takes as input two plans—which, in our case, would be $\mathcal{P}$ and $\mathcal{C}$—identifies all the threats between them, constructs a constraint satisfaction problem (CSP) representing the threat resolution alternatives, and then solves the CSP to find some set of resolutions. This set of resolutions is equivalent to the constraint set $\mathcal{O}''$ in our Definition 5 of strong compatibility, so that a solution to the CSP guarantees strong compatibility between $\mathcal{P}$ and $\mathcal{C}$. If the CSP has no solution, so that strong compatibility fails, the algorithm indicates this by returning a failure value, and, in the current treatment, the new option is rejected. Again, this immediate rejection of incompatible options is an oversimplification, but one that we adhere to in this paper. Note that the COMBINE algorithm was not designed for temporally grounded plans, but we are developing a generalization of it that can handle such plans [17].

Our focus in this paper is on the second step in the reasoning process, in which the agent computes the cost estimate for plan $\mathcal{P}$ in context $\mathcal{C}$. Given an option $\mathcal{P}$ that is compatible with $\mathcal{C}$, we might next appeal to the OPTIMAL MERGE algorithm [22], which uses a dynamic programming approach to find an optimally

merged plan, that is, one with minimum cost. This cost is precisely $\kappa(\mathcal{P} \cup \mathcal{C})$, which could then be combined with an exact value for $\kappa(\mathcal{C})$ to yield an exact value for $\kappa(\mathcal{P}/\mathcal{C})$. The option $\mathcal{P}$ could then be accepted or rejected depending on the relation between $\kappa(\mathcal{P}/\mathcal{C})$ and $\beta(\mathcal{P})$.

In general, however, it may not be necessary to compute the exact value of $\kappa(\mathcal{P}/\mathcal{C})$; instead, as suggested earlier, an agent may be able to accept or reject a new option only on the basis of an interval estimate of its cost in context. The remainder of this section develops an algorithm to implement this idea—evaluating a new option by estimating its cost in context, and then progressively refining the estimates where necessary. Such an approach may prove to be efficient if, as we suspect, it can frequently terminate in realistic cases without the need to compute an exact cost in context. In addition, the algorithm presented here displays anytime performance, producing cost estimates of monotonically increasing accuracy. Thus, if the agent "runs out of time" in evaluating an option, and is forced to a decision before shrinking the cost range sufficiently, the agent can at least make an informed decision; it can determine, for example, how much it stands to lose. Finally, by reasoning about incomplete plans and estimated costs, the algorithm matches our intuitions about deliberation in dynamic environments.

Stepsets

Given our current restriction to complete plans, the only factor influencing plan cost is step merging. To support reasoning about possible step merges, we therefore introduce the notion of a stepset. A stepset clusters steps in a plan that share the same type.

Definition 13 (Stepset of a plan) Where $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ is a consistent plan, a *stepset* for $\mathcal{P}$ is a partition $M = \{[S_1], \ldots, [S_n]\}$ of $\mathcal{S}$ subject to the restriction that $Type(S_i) = Type(S_j)$ for any steps S_i and S_j belonging to the same equivalence class $[S_k]$.

Because type-equivalence is a necessary but not sufficient condition for placing two steps in the same equivalence class, a given plan may have several different stepsets. Consider a plan $\mathcal{P}$ with only two steps, both of which are the same type. Plan $\mathcal{P}$ will have one stepset in which its two steps are clustered, and another in which they are not. Intuitively, the former corresponds to all schedules for $\mathcal{P}$ in which its two steps are merged, while the latter corresponds to those in which they are not. In fact, the temporal constraints in $\mathcal{P}$ may prevent the merging of its two steps; that is, there may be no schedules for $\mathcal{P}$ that merge its steps. In that case, we will say that the stepset that places the two steps in same equivalence class is not feasible.

Let us make these notions precise. We will say that a schedule for a plan $\mathcal{P}$ *corresponds* to a particular stepset for $\mathcal{P}$ whenever the step merges determined by the schedule agree with those of the stepset. Let $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$ be a plan, and let M be a stepset for $\mathcal{P}$ and $\overline{\mathcal{O}}$ be a schedule for $\mathcal{P}$. Then $\overline{\mathcal{O}}$ *corresponds to* M provided that for any steps S_i and S_j in S, S_i and S_j are in the same equivalence class in M if and only if $\overline{\mathcal{O}} \vdash t_i = t_j$.

As already noted, not every stepset will have a schedule that corresponds to it. Let $M = \{[S_1], \ldots, [S_n]\}$ be a stepset for the plan $\mathcal{P} = \langle \mathcal{S}, \mathcal{O}, \mathcal{L} \rangle$. Then we define the *stepset constraints* associated with M—written, $Const(M)$—as that set containing $t_i = t_j$ whenever $[S_i] = [S_j]$, and $t_i \neq t_j$ whenever $[S_i] \neq [S_j]$. The stepset M is then defined as *feasible* just in case the plan $\langle \mathcal{S}, \mathcal{O} \cup Const(M), \mathcal{L} \rangle$ is complete and schedulable.

Stepsets represent only decisions about which steps in a plan are to be merged, neglecting any other information about the order of steps or the exact times of their performance. On the other hand, stepsets do capture all the information that is necessary for computing plan cost: since cost depends only on step merging, all of the scheduled plans that correspond to a particular stepset will have the same cost. Given a

stepset M, we can therefore introduce a notion of *stepset cost* for M, written $SSCost(M)$, as follows:

$$SSCost(M) = \sum_{[S_i] \in M} Cost(S_i).$$

The stepset cost for M represents the cost of any scheduled plan corresponding to M.

The stepsets based on a plan $\mathcal{P}$ can be organized into a lattice, as follows. The top element of the lattice is the minimally merged stepset $Minmerge(\mathcal{P})$, defined as the partition $\{[S_1], \ldots, [S_n]\}$ in which each equivalence class $[S_i]$ is identified with the unit set $\{S_i\}$. The bottom element of the lattice is the maximally merged stepset $Maxmerge(\mathcal{P})$, defined as the partition $\{[S_1], \ldots, [S_n]\}$, in which each equivalence class $[S_i]$ is identified with the set $\{S_j : Type(S_j) = Type(S_i)\}$ containing all steps sharing the type of S_i. We can then define one stepset as below another in the lattice if it results from increased merging. More exactly, where M and M' are elements of the lattice, we define $M \leq M'$ just in case: for each $[S_i]$ in M' there is an $[S_j]$ in M such that $[S_i] \subseteq [S_j]$. We can then define the *down successors* of a stepset M as those stepsets that are below M in the lattice and contain exactly one fewer member; and we can define the *up successors* of M as those stepsets that are above M in the lattice and contain exactly one more member. The down successors of M are those stepsets that can be obtained from M by merging two of its members, and the up successors of M are those stepsets from which M can be obtained through the merge of two members.

We can assume that these various stepset concepts are implemented as the following functions: SSCOST(M) calculates the stepset cost of a stepset M; MINMERGE($\mathcal{P}$) and MAXMERGE($\mathcal{P}$) form the minimally and maximally merged stepsets of the plan $\mathcal{P}$; DOWN-SUCCESSORS(M) and UP-SUCCESSORS(M) return the down successors and up successors of the stepset M in the relevant lattice; and FEASIBLE($M, \mathcal{P}$) determines whether M is a feasible stepset for plan $\mathcal{P}$. All but the last of these functions are trivial. Feasibility requires checking whether $\langle \mathcal{S}, \mathcal{O} \cup Const(M), \mathcal{L} \rangle$ is complete and schedulable. We do this by casting the problem as a constraint satisfaction one, in which the constrained variables are the steps in $\mathcal{S}$, and their domains are moments of execution: that is, a solution to the CSP problem consists of an assignment of a time point to each step. Two sets of constraints must be observed: the temporal constraints in $\mathcal{O} \cup Const(M)$, and the threat-avoidance constraints that derive from $\mathcal{L}$. Although in the worst case, solving a CSP is computationally intractable, there are a number of powerful techniques that are known to work very well in practice, and in fact, have recently been applied to large planning and scheduling problems [5, 11, 12, 4].

The algorithm

We now present our algorithm, depicted in Figure 1, for evaluating an option $\mathcal{P}$ in the context $\mathcal{C}$, under the assumption that the two plans are strongly compatible. The algorithm works with two stepset lattices, based on the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$. In a fashion somewhat reminiscent of the candidate-elimination algorithm [14], our algorithms maintains, for each lattice, an upper frontier containing the highest nodes in the lattice not yet known to be infeasible, and similarly a lower frontier. It then systematically attempts to establish the feasibility or infeasibility of the nodes in the frontiers, refining the cost estimates for the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$, and using them to update the derived cost estimate for $\mathcal{P}$ in the context $\mathcal{C}$. After each refinement, the derived estimate of cost in context is compared with the benefit of $\mathcal{P}$; if $\beta(\mathcal{P})$ is outside the range of the current estimate, the algorithm then terminates with a recommendation to either accept or reject $\mathcal{P}$.

In more detail, the algorithm begins by calling the procedure INITIALIZE($\mathcal{P}, \mathcal{C}$), which has the following effects. Where $\mathcal{Q}$ ranges over the plans $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$, it assigns to the variables $Upper_{\mathcal{Q}}$ and $Lower_{\mathcal{Q}}$ sets containing the minimally and maximally merged stepsets for $\mathcal{Q}$. Next, the variables $\epsilon^+_{\mathcal{Q}}$ and $\epsilon^-_{\mathcal{Q}}$, representing upper and lower cost estimate bounds for the plan $\mathcal{Q}$, are assigned the stepset costs of these minimally and

```
procedure EVALUATE-OPTION(P, C) return Accept or Reject
    INITIALIZE(P, C)
    loop
        if COHERENT(ε⁺_C, ε⁻_C, ε⁺_{P∪C}, ε⁻_{P∪C}) then
            ε⁻_{P/C} ← max[0, ε⁻_{P∪C} − ε⁺_C]
            ε⁺_{P/C} ← ε⁺_{P∪C} − ε⁻_C
            if v[P] > ε⁺_{P/C} then
                return Accept
            end if
            if v[P] < ε⁻_{P/C} then
                return Reject
            end if
            if ε⁻_{P/C} = ε⁺_{P/C} then
                return Accept or Reject
            end if
        end if
        Call either REFINE(C) or REFINE(P ∪ C)
    end loop
```

Figure 1: EVALUATE-OPTION$(\mathcal{P}, \mathcal{C})$

maximally merged stepsets. Finally, the variables $Actual^+_{\mathcal{Q}}$ and $Actual^-_{\mathcal{Q}}$ are set to false: these are simply flags indicating whether the upper and lower cost estimate bounds for the plan $\mathcal{Q}$ are based on stepsets known to be feasible.

After initialization, the algorithm enters its main loop, first checking whether the current cost estimates for $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$ are jointly coherent, that is, whether $\epsilon^-_{\mathcal{C}} \leq \epsilon^-_{\mathcal{P}\cup\mathcal{C}}$ and $\epsilon^+_{\mathcal{C}} \leq \epsilon^+_{\mathcal{P}\cup\mathcal{C}}$. This is required because subsequent refinement of the estimates $\epsilon_{\mathcal{C}}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ may lead to a temporary loss of joint coherence. In that case, the estimate $\epsilon_{\mathcal{P}/\mathcal{C}}$ cannot be updated until coherence has been restored by further refiment steps. When $\epsilon_{\mathcal{C}}$ and $\epsilon_{\mathcal{P}\cup\mathcal{C}}$ are jointly coherent, the algorithm applies Definition 12 to update the estimate $\epsilon_{\mathcal{P}/\mathcal{C}}$, and then carries out the reasoning procedure described at the end of Section 3: it compares the benefits of the new option with the current estimate of its cost in context, and accepts or rejects the new option if the current estimate is narrow enough to allow it to do so. Otherwise, if the current estimate of cost in context is not narrow enough to justify a decision—or if the current cost estimates for $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$ are not jointly coherent—the algorithm tightens the cost estimates for $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$, by calling either REFINE$(\mathcal{C})$ or REFINE$(\mathcal{P} \cup \mathcal{C})$, and then repeats the loop.

We have left the decision about which estimate to refine ($\epsilon_{\mathcal{C}}$ or $\epsilon_{\mathcal{P}\cup\mathcal{C}}$) nondeterministic in the algorithm, thereby decoupling the control heuristics from the algorithm itself. A similar decoupling has proven to be useful in the analysis of many AI algorithms. This tactic has two advantages. First, it simplifies the base algorithm; see [20] for an example of this applied to classical plan generation. Second, it allows for separate, focused investigation of heuristics for efficient control, something we are now exploring in our ongoing work. Recent work in decision-theoretic plan generation [6] has analyzed the use of interval-based utility estimates in finding optimal plans, and there is potential for the transfer of this analysis to our framework. However, we defer consideration of this issue to subsequent work.

The procedure REFINE$(\mathcal{Q})$—where $\mathcal{Q}$ ranges over $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$—is itself straightforward, simply calling either REFINE-DOWN$(\mathcal{Q})$ or REFINE-UP$(\mathcal{Q})$, again, nondeterministically choosing between them.

```
procedure REFINE-DOWN(Q)
    if Actual⁻_Q = F then
        Select a maximal cost element M from Upper_Q
        Delete M from Upper_Q
        Add DOWN-SUCCESSORS(M) to Upper_Q
        if FEASIBLE(M, Q) then
            Actual⁺_Q ← T
            ε⁺_Q ← SSCOST(M)
        else
            if Actual⁺_Q = F then
                ε⁺_Q ← Argmax_{U ∈ Upper_Q} SSCOST(U)
            end if
        end if
    end if
```

Figure 2: REFINE-DOWN($\mathcal{Q}$)

The two refinement procedures are similar, and we first describe REFINE-DOWN($\mathcal{Q}$). Essentially, what it does is select a node M from the upper frontier, and check its feasibility. The nodes are selected in decreasing order of cost. Thus, if M is found to be feasible, its cost is less than that of any previously examined feasible node; hence the current upper bound on the cost estimate, $\epsilon^+_{\mathcal{Q}}$, can be set to the cost of M. What if M is found to be infeasible? Both $\mathcal{C}$ and $\mathcal{P} \cup \mathcal{C}$ are consistent; the latter is a consequence of the strong compatibility of $\mathcal{P}$ and $\mathcal{C}$. Thus, regardless of whether $\mathcal{Q}$ is $\mathcal{C}$ or $\mathcal{P} \cup \mathcal{C}$, its stepset lattice is guaranteed to contain at least one feasible node. If a feasible node has not yet been found—that is, if $Actual^+_{\mathcal{Q}} = F$—we know that some feasible element exists whose stepset cost is less than that of M, and so $\epsilon^+_{\mathcal{Q}}$ can be set to reflect the highest stepset cost of the remaining elements. Again, this can only lower the value of $\epsilon^+_{\mathcal{Q}}$. However, if a feasible point has already been found, we have no guarantee of finding another, and so determining that M is infeasible does not lead to updating the value of $\epsilon^+_{\mathcal{Q}}$.

After examining M, the procedure removes it from the upper frontier, and replaces it with its down-successors, so that they can be considered in subsequent iterations.

The procedure REFINE-UP($\mathcal{Q}$), presented in Figure 3, is in some ways a dual to REFINE-DOWN($\mathcal{Q}$): it refines the lower bound $\epsilon^-_{\mathcal{Q}}$ of the cost estimate for the plan $\mathcal{Q}$, working from the lower frontier. It differs, however, in its behavior when a feasible node is found. The reason for this is that the first feasible node found represents $\kappa(\mathcal{Q})$, the true cost of $\mathcal{Q}$: because the procedure selects nodes in increasing order of cost, the first feasible node it finds necessarily corresponds to the least expensive schedule for $\mathcal{Q}$, which is the true cost of $\mathcal{Q}$. Therefore, immediately upon finding a feasible node, REFINE-UP($\mathcal{Q}$) sets the upper bound $\epsilon^+_{\mathcal{Q}}$ to $\epsilon^-_{\mathcal{Q}}$, narrowing the estimate to a point.

One final comment about the algorithms concerns the initial lines of both REFINE-UP($\mathcal{Q}$) and REFINE-DOWN($\mathcal{Q}$), which require that $Actual^-_{\mathcal{Q}} = F$. This is simply a bookkeeping condition: because REFINE-UP($\mathcal{Q}$) sets both $\epsilon^-_{\mathcal{Q}}$ and $\epsilon^+_{\mathcal{Q}}$ to the exact cost of $\mathcal{Q}$ at the same time as it sets $Actual^-_{\mathcal{Q}} = T$, this condition blocks additional calls to either procedure from producing any further effect.

```
procedure REFINE-UP(Q)
    if Actual_Q^- = F then
        Select a minimal cost element M from Lower_Q
        Delete M from Lower_Q
        Add UP-SUCCESSORS(M) to Lower_Q
        if FEASIBLE(M, Q) then
            Actual_Q^- ← T
            ε_Q^- ← SSCOST(M)
            ε_Q^+ ← ε_Q^-
        else
            ε_Q^- ← Argmin_{U ∈ Lower_Q} SSCOST(U)
        end if
    end if
```

Figure 3: REFINE-UP($\mathcal{Q}$)

5 Conclusion

In this paper, we have developed the foundations of a theory of rational choice that takes seriously the view that agents make decisions in the context of their existing plans. A central goal of our work is to ensure that the theory we develop is computationally viable, and we have framed our approach in terms of models of planning developed in the AI literature. The overall picture we have is one in which an agent maintains a set of commitments, some of which, at any given time, will be only partially specified. A new option for action must then be evaluated taking into account interactions between that option and the background context. Recognizing that agents have computational resource limits, we have developed an account that does not require the agent always to compute the exact cost of an option in context; instead, we have shown how estimates of option cost can support rational choice.

Here we have only provided details for a very restricted case of the general problem, and it is clear that the theory must be extended along a number of dimensions. Some such extensions are relatively straightforward. For example, we have discussed only cost savings that result from merging of type-identical steps, but sometimes it is also possible to merge steps of different types that both achieve the particular conditions required in the current context. Also, we have discussed only plans that involve a single, albeit possibly merged, action at a time, but have not here considered plans with true concurrency: two or more steps of different types performed at the same moment.

Other generalizations will require more significant extensions to the theory as we have developed it in the paper. Most notably, we have dealt here only with complete, primitive plans. Handling incomplete plans requires extending the algorithms for cost computation, because the cost of an incomplete plan depends not just on possible ways of scheduling it, but also on possible ways of completing it. Similarly, handling hierarchical plans requires reasoning about possible decompositions. In both cases, then, methods for plan generation must be interwoven with the cost computation process.

Finally, we have focused our attention in this paper on new options that are at least strongly compatible with the background context. Although this is a reasonable starting point, it is clearly important to generalize the theory so it can handle other situations. We are currently examining weaker notions of compatibility, for instance, situations in which the union of the new option and the background context results in threats that cannot be resolved with just ordering and linking constraints, but instead require the introduction of new

plan steps. In this case, the computation of plan cost must take into account the cost of the threat-repair steps. A further generalization, hinted at in Section 3, would cover situations of true incompatibility between a background context and a new option.

References

[1] Fahiem Bacchus and Froduald Kabanza. Planning for temporally extended goals. In *Proceedings of the 13th National Conference on Artificial Intelligence*, 1996.

[2] Michael E. Bratman. *Intention, Plans and Practical Reason.* Harvard University Press, Cambridge, MA, 1987.

[3] Michael E. Bratman, David J. Israel, and Martha E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:349–355, 1988.

[4] Amedeo Cesta, Angelo Oddi, and Stephen F. Smith. Profile-based algorithms to solve multiple capacitated metric scheduling problems. In *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS-98)*, pages 214–223, 1998.

[5] C. Cheng and S. F. Smith. Generating feasible schedules under complex metric constraints. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, 1994.

[6] Richard Goodwin and Reid Simmons. Search control of plan generation in decision-theoretic planners. In *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS-98)*, pages 94–101, 1998.

[7] P. Haddawy, A. Doan, and R. Goodwin. Efficient decision-theoretic planning: techniques and empirical analysis. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, 1995.

[8] Ronald A. Howard and James E. Matheson, editors. *The Principles and Applications of Decision Analysis.* Strategic Decision Group, Menlo Park, CA, 1984.

[9] Daniel Kahneman and Amos Tversky. Prospect theory: an analysis of decision under risk. *Econometrica*, 47:263–291, 1979.

[10] Subbarao Kambhampati, Craig A. Knoblock, and Qiang Yang. Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence*, 76(1-2):167–238, 1995.

[11] Henry Kautz and Bart Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI)*, Portland, OR, 1996.

[12] Henry Kautz and Bart Selman. The role of domain-specific knowledge in the planning as satisfiability framework. In *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS-98)*, pages 181–189, 1998.

[13] David McAllester and David Rosenblitt. Systematic nonlinear planning. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 634–639, Anaheim, CA, 1991.

[14] Tom M. Mitchell. *Machine Learning.* McGraw-Hill Co., Inc., New York, 1997.

[15] J. Scott Penberthy and Daniel Weld. UCPOP: A sound, complete, partial order planner for ADL. In *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, pages 103–114, Cambridge, MA, 1992.

[16] Martha E. Pollack. The uses of plans. *Artificial Intelligence*, 57:43–68, 1992.

[17] Martha E. Pollack, Ioannis Tsamardinos, Jun Hu, and John F. Horty. Merging temporally grounded plans. In preparation.

[18] Arthur Prior. *Past, Present and Future*. Oxford University Press, New York, 1967.

[19] Richmond Thomason. Indeterminist time and truth-value gaps. *Theoria*, 36:264–281, 1970.

[20] Daniel S. Weld. An introduction to least commitment planning. *AI Magazine*, 15(4):27–61, 1994.

[21] Mike Williamson and Steve Hanks. Optimal planning with a goal-directed utility model. In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*, pages 176–181, 1994.

[22] Qiang Yang. *Intelligent Planning: A Decomposition and Abstraction Based Approach*. Springer, New York, 1997.