Applying Decision Theory to Reactive Planning*

Fahiem Bacchus
Dept. of Computer Science
University of Waterloo
Waterloo, Ontario, Canada, N2L-3G1
fbacchus@logos.uwaterloo.ca

Froduald Kabanza
Dept. de Math et Informatique
Universite de Sherbrooke
Sherbrooke, Quebec, Canada, J1K-2R1
kabanza@dmi.usherb.ca

1 Reactive Planning

We are engaged in a research project aimed at the synthesis of *reactive plans* [BD90, GK91, Kab90, Mit90, DKKN93a]. Reactive plans are different from classical plans in that they are specified as a collection of rules or "reactions" of the form $S \Rightarrow a$, where S is a description (perhaps partial) of a situation or state and a is an action executable by the agent (such rules have also been called situated control rules by Drummond [Dru89]). The meaning of such a rule is that if the agent finds itself in a state characterized by S, then it should execute action a.

Hence, a key difference between classical plans and reactive plans is that reactive plans contain no notion of a predetermined action execution sequence. Rather, the sequence of actions executed will depend on what happens in the agent's environment. This makes reactive plans more successful in real environments where the agent's actions are not the only source of change. Another key difference is that a classical plan, i.e., a finite sequence of actions to be executed by the agent, can only achieve a limited set of goals. For the most part only simple goals of achievement have been studied in this context. As we will discuss in more detail below, a reactive plan can be used to achieve certain types of temporally extended goals that are beyond the capability of simple classical plans.

Unless we take the approach of universal plans [Sch87] where we store a reaction for every possible situation, we need to be able to compute reactions dynamically. The ability to do *reactive planning* in this way has a number of benefits. First, it allows the agent to adapt to its environment and to changes in its environment by replacing its reactions with newly computed superior reactions. Second, the reactive planner can easily provide an anytime capability by always retaining its old reactions until it can replace them with better ones. And third, a reactive planner allows us to specify new goals to the agent and have it alter its reactions so as to achieve these new goals.

There are three main components of the planning architecture we are investigating: (1) a theory of action, (2) a language for expressing goals, and (3) our approach to planning. In the next three sections we present some additional details about each of these components, contrasting our approach to previous work.

2 Actions

The fact that the agent is not the only source of change in its environment is a key motivation behind reactive planning. The execution of reactive plans involve the continual sensing of the environment by the agent. At each step the agent senses its environment and executes the particular action specified by its collection of reaction rules. These actions affect the environment, but not in a completely predictable manner. The uncertainty of the action effects is due to both random errors on the part of the agent and random exogenous events occurring in the environment. The aim of the agent's actions, then, is to influence, rather than determine, changes in its environment, changes that serve to achieve the agent's goals.

Planning in this domain requires a theory of action that can project the future evolution of the environment under various choices of agent reactions. To model uncertainty, probabilities can be assigned to the various possible state evolutions. Searching for a plan then becomes searching for a set of agent reactions that influences the environment in a manner that best satisfies the agent's goals.

A satisfactory theory of action must be able to efficiently project the components of the world state that are relevant to the agent's goals. For example, human agents have rich, multi-faceted, representations of the world state. To project what will happen to all of these aspects of the world would be very difficult. Nevertheless, when planning we can often efficiently project those aspects of the world state relevant to our particular goal. Efficient temporal projection is essential for planning, as searching for a plan requires projecting the evolution of the current state under many different candidate plans.

To our knowledge, a theory of action that can deal well with exogenous events has yet to be developed. Dean et al. [DKKN93a] and also Drummond and Bresina [DB90] both assume access to the end product of a theory of action, i.e., an efficient encoding (via a stochastic automaton) of the probabilistic effects of each of the agent's actions in every state. With such information planning can

*This work has been supported by the Canadian Government through their IRIS program.

be performed using either dynamic programming algorithms or search: the system determines the best action to be executed in each state relying on the fact that it knows the probabilistic effects of every action in every state.

The theory of action developed by Hanks [Han90] is an extension of the standard STRIPS theory of action, where actions are specified by add and delete lists. In Hank's representation the effects of an action are keyed by disjoint sets of preconditions which include chance events. Given a STRIPS-style state description the successor states arising from the execution of an action can be computed by matching the appropriate preconditions. When chance events are included in the preconditions, multiple successor states are generated and each one is assigned the probability of the chance event. Hence, a collection of actions so specified implicitly defines a stochastic automaton, where the probabilistic effects of each action in each state are computed simply by generating the probabilistically labeled successor states. With this stochastic automaton the algorithms of [DB90, DKKN93a] can be applied to generate appropriate agent reactions.

Although it has the advantage of supporting efficient temporal projection, this form of action representation has difficulties when dealing with exogenous events.

- It associates changes in the environment solely with actions of the agent. In a dynamic environment the world is changing even if the agent does not perform an action. One option is to introduce a huge *wait* action. Such an action would, using exclusive precondition sets, characterize the effects of all of the exogenous events that can occur in various situations given that the agent does nothing.
- It does not allow for exogenous events that occur in parallel with the actions of the agent. To a certain extent the parallel effects of exogenous events are encoded within the probabilistic actions. That is, one way of viewing the multiple possible outcomes of an action is that these are the various outcomes of the action that occur when various exogenous events occur during their execution.
- Our model of the environment might be too complex to be taken into account in the specification of agent's actions. This is the key difficulty in folding exogenous events into the agent's actions, as in the solutions suggested to the two problems above. For each particular goal there are various aspects of the world that we do not need to worry about. Hence, we want to consider the effects of exogenous events only on those aspects of the world that affect the attainment of the agent's goals. By folding exogenous events into our specification of agent actions we make it more difficult to selectively project only certain aspects of the world.

We are currently working on a theory of action that maintains a separation between the agent's actions and exogenous events. Our theory maintains a complex model of exogenous events which can be selectively projected, i.e., we can project certain aspects of the world while ignoring others. We are working on a theory of causal interaction that would allow us to project the interactive effects of the exogenous events and the agent's actions. However, at the time of writting this document a number of choices remain to be settled, so we cannot report any further details.

3 Expressing Goals

3.1 Temporally Extended Goals

One of the most important features of our approach is the use of temporal logics to represent the agent's goals. Various temporal logics have been developed for the purpose of program specification and verification, and much of the work in this field can be usefully applied to work in planning.

Classical planning has mainly addressed the achievement of *static* goals, i.e., goals that are satisfiable in a single state. Under this restriction, a plan to achieve a goal simply needs to attain a final state in which the goal is satisfied; the states the agent passes through during the execution of the plan play no role in determining the plan. In contrast, *temporally extended* goals cannot be satisfied by a single state. Rather, such goals describe properties of *sequences* of states. The desirability of planning for certain classes of temporally extended goals has been recognized for some time in the literature. For example, *maintenance* goals, goals of *prevention*, and goals with *deadlines* have all been discussed [HH93, Dru89, DKKN93b, Ver83], and some approaches have been proposed for planning for such goals.

Temporal logics (see, e.g., [Eme90] for a overview) allow the expression of a wide variety of temporally extended goals. These languages are, typically, interpreted over a sequence of states, hence the states the agent passes through on its way to a final state play a role in determining whether or not a goal expressed in these languages is satisfied. Even more importantly in the context of reactive plans, is that there is no final state that is the result of executing a reactive plan. Instead reactive plans specify an ongoing interaction between the agent and its environment. Hence, the classical notion of static goal achievement does not make much sense in this context. Temporal logics can be used to characterize various properties of these ongoing interactions.

Important among these properties are the so called "liveness" properties. These are properties that can only be satisfied by an unending sequence of states. Reactive plans, unlike classical plans, can achieve liveness conditions. Classical plans are generally restricted to being finite sequences of actions, and thus only a finite sequence of states are visited during their execution. For example, a reactive plan can achieve the goal of always turning the lights on at 10:00pm and always turning them off at 10:00am. There is no classical plan for achieving this goal: such a plan would require an infinite sequence of wait-turnon-wait-turnoff actions.

Another important benefit of using these logics to express temporally extended goals is that they allow the declaratively expression the agent's goals in a notation that has a clear and unambiguous semantics. This is vitally important if we ever want to verify that the agent is "doing the right thing:" we must have an unambiguous expression of what the right thing is before we can determine if the agent is doing it!

3.1.1 TPTL

For the purposes of dealing with reactive planning we are employing a modified version of *timed propositional temporal logic* (TPTL) developed by Alur and Henzinger [AH89]. This logic allows to express goals with deadlines as well ordinary prevention and maintenance goals. We turn now to a presentation of this logic, adopted from [Hen91].

Syntactically, we start with a finite set of propositional symbols P and a set of temporal variables V. The terms π and formulas ϕ of TPTL are generated by the inductive rules:

$$\begin{aligned}\pi &:= x \mid x+c \mid c.\\ \phi &:= p \mid \pi_1 \leq \pi_2 \mid \neg\phi_1 \mid \phi_1 \wedge \phi_2 \mid \bigcirc\phi_1 \mid \phi_1 \mathsf{U} \phi_2 \mid x.\phi_1\end{aligned}$$

where $x \in V$, $p \in P$, and c is a natural number. The other equality relations and boolean connectives can be defined as abbreviations. Furthermore, we can define the following additional temporal operators:

Eventually: $\Diamond\phi =_{\text{def}}$ TRUE $\mathsf{U}\, \phi$.

Always: $\Box\phi =_{\text{def}} \neg\Diamond\neg\phi$.

Semantically, formulas of TPTL are interpreted over timed state sequences. That is, we have a sequence of states $\sigma = \sigma_0, \sigma_1, \ldots$, where each state σ_i is identified with the set of propositions $p \in P$ true in that state, and has an associated integral time stamp T_i.[1] Let $\rho = (\sigma, \mathsf{T})$ be a sequence of states $(\sigma_0, \sigma_1, \ldots)$ with associated time stamps $(\mathsf{T}_0, \mathsf{T}_1, \ldots)$. Let ρ^i denote ρ with the first i states of σ and time stamps of T removed. Let $\mathcal{E} : V \to I\!N$ be an interpretation for the variables V, i.e., a mapping that assigns a natural number to each variable. The pair $(\rho, \mathcal{E})$ satisfies the TPTL-formula ϕ iff $\rho \models_{\mathcal{E}} \phi$, where "$\models$" is defined inductively as follows for all $i \geq 0$:

- $\rho^i \models_{\mathcal{E}} p$ iff $p \in \sigma_i$.
- $\rho^i \models_{\mathcal{E}} \pi_1 \leq \pi_2$ iff $\mathcal{E}(\pi_1) \leq \mathcal{E}(\pi_2)$.
- $\rho^i \models_{\mathcal{E}} \neg\phi$ iff $\rho^i \not\models \phi$.
- $\rho^i \models_{\mathcal{E}} \phi_1 \wedge \phi_2$ iff $\rho^i \models_{\mathcal{E}} \phi_1$ and $\rho^i \models_{\mathcal{E}} \phi_2$.
- $\rho^i \models_{\mathcal{E}} \bigcirc\phi$ iff $\rho^{i+1} \models_{\mathcal{E}} \phi$.
- $\rho^i \models_{\mathcal{E}} \phi_1 \mathsf{U} \phi_2$ iff for some $j \geq i$, $\rho^j \models_{\mathcal{E}} \phi_2$, and for all $i \leq k < j$, $\rho^k \models_{\mathcal{E}} \phi_1 \wedge \neg\phi_2$.[2]
- $\rho^i \models_{\mathcal{E}} x.\phi$ iff $\rho^i \models_{\mathcal{E}[x:=\mathsf{T}_i]} \phi$.

Here, $\mathcal{E}(x+c) = \mathcal{E}(x) + c$, $\mathcal{E}(c) = c$, and $\mathcal{E}[x := t]$ is the same as $\mathcal{E}$ except that it maps x to $t \in I\!N$.

3.1.2 Examples

The semantics of TPTL are quite subtle at first but a couple of examples and some practice soon makes it fairly straightforward.

- $\Box x.(p \Rightarrow \Diamond y.q \wedge y \leq x+10)$. This asserts a property of *bounded response*. Specifically, it says that at every state σ_i (where x is now bound to the time stamp of that state T_i) if p is true then q must be achieved in some future state σ_j (where y is now bound to the time stamp of that state T_j) such that $y \leq x + 10$, i.e., we must have $\mathsf{T}_j \leq \mathsf{T}_i + 10$. In other words, whenever p becomes true, ensure that q is achieved within ten time units.
- $\Box x.(p \Rightarrow p \,\mathsf{U}\, (y.y \geq x + 10))$. This formula asserts a property of *bounded maintenance*. Specifically, it says that whenever p becomes true it must remain true until we reach a state that is at least ten time units into the future.
- $\Box x.(p \Rightarrow p \,\mathsf{U}\, (y.(q \wedge y \leq x + 10)))$. This formula asserts a mixed property of bounded achievement and maintenance. Specifically, it says that whenever p becomes true, it must be maintained until q is achieved and that achievement must be within 10 time units.

Two things are important to notice. First, we can specify arbitrarily complex goals, but the semantics of these goals remains reducible to simple semantic constraints. And second, these kinds of triggered goals (triggered by p in our examples) are not easy to express using the reward function formalism [DKKN93b], nor are they easy to express using the language of [HH93] due to the strong separation between propositions and time in that language.[3]

3.1.3 Our Modification of U

The semantic definition we gave above for the temporal operator U is not standard. The standard definition (as used, e.g., in [AH89]) is as follows:
$\rho^i \models_{\mathcal{E}} \phi_1 \mathsf{U} \phi_2$ iff for some $j \geq i$, $\rho^j \models_{\mathcal{E}} \phi_2$, and for all $i \leq k < j$, $\rho^k \models_{\mathcal{E}} \phi_1$.
Where our requirement that σ^j be the first state that satisfies ϕ_2 has been removed. In the absence of temporal quantification the two definitions are identical, but in their presense our definition is more restrictive. For example, consider the state sequence $\sigma_0, \sigma_1, \sigma_2, \sigma_3$, where $\sigma_0 \models p \wedge \neg q$, $\sigma_1 \models p \wedge \not q$, $\sigma_2 \models p \wedge q$ and $\sigma_3 \models \neg p \wedge q$. Under our definition we have the sequence starting at σ_0 satisfies $x.p \,\mathsf{U}\, (y.q \wedge y = x + 2$, but does not satisfy $x.p \,\mathsf{U}\, (y.q \wedge y = x + 3)$, whereas the standard definition satisfies both formulas. In other words, our definition forces a "first-state" interpretation of U. This turns out to be important when we turn to assigning utilities to goals.

3.2 The Utility of Goals

Once we accept a probabilistic model of action effects is seems only natural to generalize our notion of goal achievement to a utility based formalism. In the utility based formalism the agent is no longer given a "goal," rather it is given a collection "desirabilities," i.e., things that are desired to various degrees. Now the goal of the

[1]The restriction to integral time stamps simply means that we are dealing with a discrete model of time.

[2]The specification of U is different from the standard semantics. See the text below for an explanation.

[3]This comparison is not really fair until we demonstrate how we can represent goal utilities using this formalism, but we will do this in the next section.

agent becomes the meta-level goal of achieving as many of these desirabilities as is possible. Or as specified more precisely in standard decision theory, the agent's goal is always to maximize its expected utility.

Dean et al. [DKKN93b] are dealing specifically with reactive planning, and they show how certain types of temporally extended goals can be specified using reward functions that assign utilities to *individual* states. Our goal language, on the other hand, makes the natural notion on which utilities should be specified a sequence of states, not an individual state. This is also the approach of Haddawy and Hanks [HH93], who assign utilities to *chronicles*, i.e., sequences of states corresponding to a particular course of execution of a plan.

We feel that assigning utilities to state evolutions is a more natural choice, especially in the domain of reactive planning where the plan specifies an ongoing interaction between the agent and its environment. For example, Dean et al. must resort to a principle of future discount to compute the expected utility of this ongoing interaction.

We cannot use the goal language of [HH93], as this language cannot express liveness properties that are achievable by reactive plans. Hence, to move to a decision theoretic framework we must provide a mechanism for associating utilities with state evolutions (or chronicles).

The approach we take is to associate benefits with goals. To accomplish this we write goal schema instead of goals with specific deadlines. For example, say that there is a benefit in achieving p as soon as possible. The utility assigned to a particular state evolution sequence will depend on how soon p is achieved in that sequence. To capture this we write the axiom schema

$$\Diamond x.p \wedge x = U,$$

where the various instances of this schema are generated by substituting particular numbers for the parameter U. When we evaluate the satisfaction of this goal on a particular sequence of states on which p eventually becomes true, only one particular instance of this schema will be satisfied. For example, if p is first achieved in the state whose time stamp is 10 then only the instance $\Diamond x.p \wedge x = 10$, will be satisfied.[4]

Once we have determined the value of the parameter U for which the state sequence satisfies the goal, we can use this value as input to a reward function. Hence, to specify a benefit in achieving p as soon as possible we would use a decreasing reward function. Thus, if state sequence a satisfied the goal with $U = 2$, it would be given higher utility than a state sequence b that satisfied the goal with $U = 10$.

This approach is quite general. For example, we can use it to represent utilities of triggered goals. We can use the schema

$$\Box x.(p \Rightarrow p \,\mathsf{U}\, (y.q \wedge x = y + U)),$$

and now U measures the time take to achieve p once triggered by p. If we now write the schema

$$\Box x.(p \wedge x = U_1 \Rightarrow p \,\mathsf{U}\, (y.q \wedge x = y + U_2)),$$

[4]This is where our modified semantics for U comes into play. Using the standard semantics many different instances of this formula could be satisfied by the sequence.

we can have a two parameter reward function that discounts the benefit of quick response (i.e., small values of U_2) that occur far into the future (i.e., large values of U_1).

There are two components that complete our approach to assigning utilities to goals.

1. We restrict the class of goal schema formulas so that our approach of parameterization works properly. For example, in the previous schema, if given a state sequence in which p is true from the start state and remains true for 10 time units when q is achieved, the schema would be satisfied for multiple pairs of values. For example $(U_1 = 0, U_2 = 10)$, $(U_1 = 1, U_2 = 9)$, $(U_1 = 2, U_2 = 8)$, etc., are all instances of the axiom schema satisfied by this state sequence. Clearly, we only want to use the first time that p is satisfied, i.e., $(U_1 = 0, U_2 = 10)$ in evaluating the utility of the sequence.
2. A given state sequence can satisfy many different goals. We do not want to take the common approach of forcing the goals to be mutually exclusive. Rather, we use a *ceteris paribus* assumption to compute the utility of a sequence that satisfies multiple goals.

4 Planning

Finally, a few words about our approach to planning. Essentially, it is very similar to the approach taken by Drummond and Bresina [DB90]. Specifically, it involves forward projection of the current state to generate probable sequences of state evolutions under various sets of agent reactions. Using ideas contained in [Kab92], we evaluate the goal formulas on these state sequences by a process of goal progression. Concurrent with goal progression we can calculate the sequence's utility by determining which goals the sequence satisfies and which goals the sequence violates. Our theory of action tells us how likely any particular state evolution is, and our utility computations tell us how valuable that sequence is. Together, we can use this information to evaluate the merit of candidate collections of agent reactions (by computing estimates of the expected utility generated by the probable state evolutions these reactions cause). Of course, empirical evidence will have a considerable influence on the future evolution of our approach, but our first experiments will be conducted using the approach sketched here.

References

[AH89] R. Alur and T. A. Henzinger. A really temporal logic. In *Proc. 30th IEEE Symp. on Foundations of Computer Science*, pages 164–169, 1989.

[BD90] J. Bresina and M. Drummond. Integrating planning and reaction. In *AAAI workshop on Planning in Uncertain, Unpredictable, or Changing Environments*, Stanford Univ., March 1990.

[DB90] M. Drummond and J. Bresina. Anytime synthetic projection: Maximazing probability of goal satisfaction. In *Proc. National Conference on Artificial Intelligence (AAAI '90)*, pages 138–144, 1990.

[DKKN93a] T. Dean, L. P. Kaelbling, J. Kerman, and A. Nicholson. Planning with deadlines in stochastic domains. In *Proc. National Conference on Artificial Intelligence (AAAI '93)*, pages 574–579, 1993.

[DKKN93b] T. Dean, L. P. Kaelbling, J. Kirman, and A. Nicholson. Planning under time constraints in stochastic domains. unpublished manuscript, 1993.

[Dru89] M. Drummond. Situated control rules. In *Proc. First International Conference on Principles of Knowledge Representation and Reasoning (KR '89)*, pages 103–113. Morgan Kaufmann, 1989.

[Eme90] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B*, chapter 16, pages 997–1072. MIT, 1990.

[GK91] P. Godefroid and F. Kabanza. An efficient reactive planner for synthesizing reactive plans. In *Proc. National Conference on Artificial Intelligence (AAAI '91)*, pages 640–645, 1991.

[Han90] S. Hanks. Practical temporal projection. In *Proc. National Conference on Artificial Intelligence (AAAI '90)*, pages 158–163, 1990.

[Hen91] T. Henzinger. *The Temporal Specification and Verification of Real-Time Systems*. PhD thesis, Department of Computer Science, Stanford University, 1991. available as technical report STAN-CS-91-1380.

[HH93] P. Haddawy and S. Hanks. Utility models for goal-directed decision-theoretic planners. Technical Report 93–06–04, University of Washington, 1993. Technical Report.

[Kab90] F. Kabanza. Synthesis of reactive plans for multi-path environments. In *Proc. National Conference on Artificial Intelligence (AAAI '90)*, pages 164–169, 1990.

[Kab92] F. Kabanza. *Reactive Planning of Immediate Actions*. PhD thesis, Departement d'Informatique, Universite de Liège, Belgium, October 1992.

[Mit90] T. M. Mitchell. Becoming increasingly reactive. In *Proc. National Conference on Artificial Intelligence (AAAI '90)*, pages 1051–1058, 1990.

[Sch87] M. J. Schoppers. Universal plans for reactive robots in unpredictable environments. In *Proc. Tenth International Joint Conference on Artificial Intelligence (IJCAI '87)*, pages 1039–1046, 1987.

[Ver83] S. Vere. Planning in time: Windows and durations for activities and goals. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5, 1983.