Goal specification using temporal logic in presence of non-deterministic actions

Chitta Baral
Department of Computer Sc. and Engg.
Arizona State University
Tempe, Arizona 85287
chitta@asu.edu

Matt Barry
Advance Tech Development Lab
United Space Alliance
Houston, TX 77058
Matthew.R.Barry@USAHQ.UnitedSpaceAlliance.com

Abstract

In this paper we show that despite a past claim goals such as 'try your best to make p true' in presence of non-deterministic actions can be expressed in the framework of branching time temporal logic. We analyze the A and E operators in CTL* and point out why it was thought that the above mentioned goal can not be expressed using CTL*. We then introduce the operators A_π and E_π (and present a temporal logic π-CTL*) which take into account the policy being followed by the agent and show that using these operators we can indeed specify the above mentioned goal.
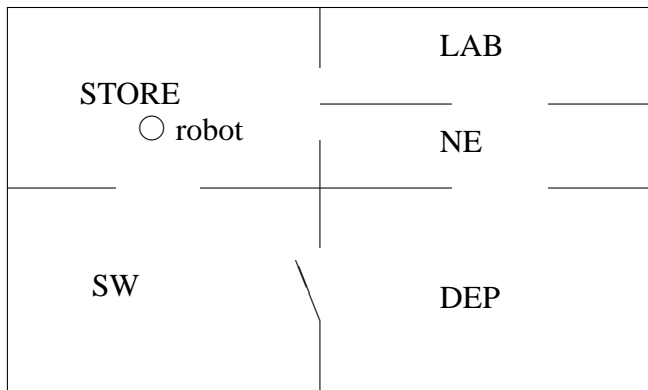
1 Introduction and motivation

Figure 1: A navigation domain

In recent years it has been proposed [Bacchus and Kabanza, 1998; Niyogi and Sarkar, 2000; Pistore and Traverso, 2001; Baral *et al.*, 2001] that temporal logic be used to specify goals that go beyond only putting conditions on the final state. Most of these papers – except [Pistore and Traverso, 2001], only consider the case when actions are deterministic. In [Dal Lago *et al.*, 2002] it is argued that in presence

of non-deterministic actions[1] certain kind of goals can not be expressed using temporal logic and an alternative language to specify goals is crafted. In this paper we counter this claim and show that some of the goals discussed there can indeed be expressed using temporal logic if we define the operators appropriately.

To motivate the difficulties associated with goal specification in presence of non-deterministic actions we start with the following example from [Dal Lago *et al.*, 2002].
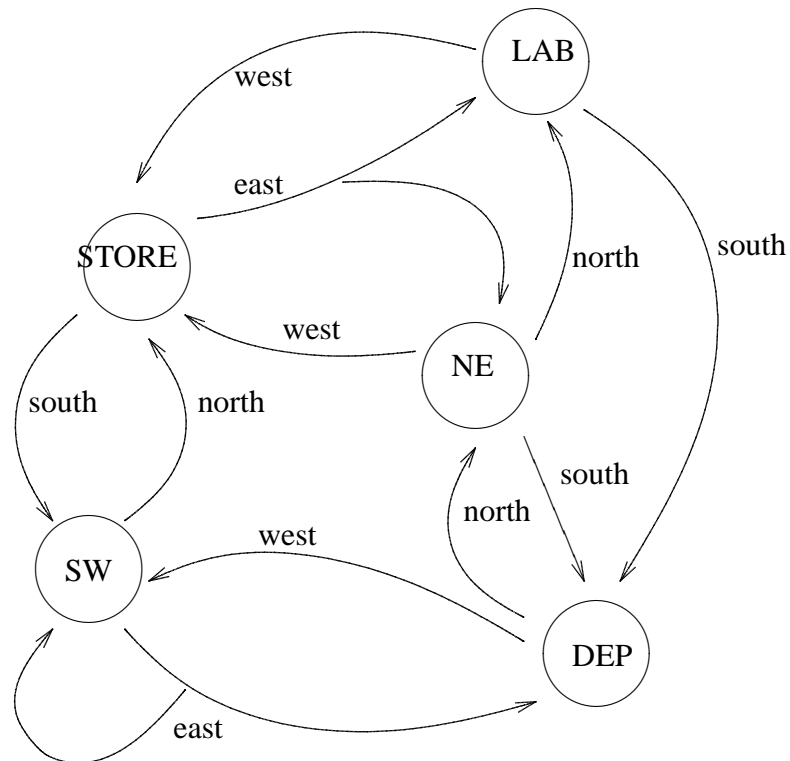

Figure 2: Transition between the locations

---

[1]Non-determinism plays an important role in modeling domains such as a space shuttle in which much of the environment cannot be sensed remotely or cannot be tested at all until acted upon.

Consider the navigation domain in Figure 1 above. In this domain we have a robot who is in the store room. The robot has actions of going west, north, south or east from the different rooms and most of its actions have intended effects as shown in the transition diagram in Figure 2 except that: (i) When it is in the STORE and it goes EAST it may either end up in the LAB or in the room NE. (ii) When it is in the room SW and it goes east it may either end up in DEP or remain in SW. This is because there is a door between SW and DEP which the robot can not sense and this door is sometimes open and sometimes closed.

Now suppose the goal of the robot (which is initially in STORE) is to try (its best) to get to DEP but avoid LAB which is a dangerous place. *How do we specify this goal, which we will refer to as goal_a?*

Before discussing the specification of this goal, let us first discuss what kind of a 'plan' will achieve this goal. There are only two ways to get to DEP: (i) going south from NE and (ii) going east from SW. To go south from NE the robot first has to get to NE. To go to NE from STORE (where the robot is initially) without going north from DEP (which does not make sense as the ultimate goal is to get to DEP) the only way is to go east from STORE. But this action is non-deterministic and may take the robot to LAB which the robot is required to avoid. Thus option (i) is ruled out. Now the only option left is is (ii). Following that a valid plan would be to go south from the initial position STORE to SW and then go east from SW. But since going east from SW is a non-deterministic action, there is no guarantee that the robot will reach DEP. But if the robot keeps continually trying to go east from SW then we can say that the robot is trying (its best) to get to DEP. Lets refer to this attempt as *plan_a*. On the other hand a robot which continually goes south from STORE and north from SW is definitely not trying to get to DEP. Lets refer to this attempt as *plan_b*.

A suitable representation of the above mentioned goal and a suitable definition of when a plan achieves a goal should be such that plan_a above would be a plan that achieves goal_a while plan_b does not.

Lets us now try to express this goal using temporal logics. A straightforward attempt to encode this in linear temporal logic with future operators (LTL) [Manna and Pnueli, 1992; Emerson, 1990] leads us to the specification ($\Box \neg LAB \wedge \Diamond \Box DEP$), where $\Box$ is the operator meaning 'always in the future' and $\Diamond$ is operator meaning 'eventually in the future'. This specification is too strong as plan_a is not a plan with respect to this goal. That is because plan_a can not guarantee that the robot will eventually get to DEP.

Now let us try to express this goal in the branching time temporal logic CTL*, where we have the operators A (meaning 'for all paths') and E (meaning 'there exists a path') at our disposal. Obviously, the representation $A(\Box \neg LAB \wedge \Diamond \Box DEP)$ is not appropriate as it requires all paths to have the property that eventually the robot would be at DEP. Similar to the LTL goal specification earlier, this specification is too strong as plan_a is not a plan with respect to this goal. Now let us consider the specification $E(\Box \neg LAB \wedge \Diamond \Box DEP)$. This specification is not appropriate either. It is weak in the sense that

it just asks the robot to have an available path to DEP and in that sense plan_b would be a valid plan with respect to this goal. As we discussed earlier this is not intended.

In [Dal Lago *et al.*, 2002] the authors present the above examples to argue that temporal logics can not represent such goals. They present an alternative specialized language with semantics defined in a completely new manner (as compared to characterization of standard temporal operators) to specify such goals.

Our goal in this paper is to stay within the framework of temporal logic and figure out a way to express these goals using temporal logic such that it is easy to combine temporally expressed goals from earlier work [Bacchus and Kabanza, 1998; Niyogi and Sarkar, 2000; Baral et al.*, 2001] with the kind of goals that we discuss in this paper.* The approach in [Dal Lago *et al.*, 2002] of using a specialized language makes such combination difficult if not impossible.

The main cornerstone of our approach is to clarify the notions A and E, which means 'for all paths', and 'for at least one path' respectively. The issue is what does 'path' mean here. (a) Is it a valid sequence of actions that is executable by the agent? (b) Is it a valid sequence of actions regardless of whether they are agent's actions or exogenous actions? (c) Is it a valid sequence of actions as dictated by the agent's plan? We will show that all these lead to different kind of A and E operators and we need all of them to express different goals. The lack of this distinction was the reason why [Dal Lago *et al.*, 2002] claimed that the goal_a is not expressible in temporal logic.

The rest of the paper is organized as follows. In Section 2 we present the existing notions of specifying goals using temporal logic and what are plans with respect to such goals. In Section 3 we discuss the notions that are necessary to express goals such as goal_a in presence of non-deterministic effects. In Section 4 we relate our proposal with some of the constructs in [Dal Lago *et al.*, 2002] and finally we conclude in Section 5.

2 Goal specification using temporal operators: past work

In this section we discuss the existing formulations [Bacchus and Kabanza, 1996; 1998; Niyogi and Sarkar, 2000; Baral *et al.*, 2001] of specifying goals using linear and branching time temporal logics, and discuss what are plans with respect to such goals. We start with goal specification using linear temporal logic.

2.1 Goal representation using LTL

Syntactically, LTL formulas are made up of propositions, propositional connectives $\vee$, $\wedge$, and $\neg$, and future temporal connectives $\bigcirc$, $\Box$, $\Diamond$ and U. We now formally define the truth value of temporal formulas with respect to trajectories.

Definition 1 Let σ given by $s_0, s_1, \ldots, s_k, s_{k+1}, \ldots$ be a trajectory, p denote a propositional formula, and f and f_is denote LTL formulas.

- $(s_j, \sigma) \models p$ iff p is true in s_j.

- $(s_j, \sigma) \models \neg f$ iff $(s_j, \sigma) \not\models f$.
- $(s_j, \sigma) \models f_1 \vee f_2$ iff $(s_j, \sigma) \models f_1$ or $(s_j, \sigma) \models f_2$.
- $(s_j, \sigma) \models f_1 \wedge f_2$ iff $(s_j, \sigma) \models f_1$ and $(s_j, \sigma) \models f_2$.
- $(s_j, \sigma) \models \bigcirc f$ iff $(s_{j+1}, \sigma) \models f$
- $(s_j, \sigma) \models \Box f$ iff $(s_k, \sigma) \models f$, for all $k \geq j$.
- $(s_j, \sigma) \models \Diamond f$ iff $(s_k, \sigma) \models f$, for some $k \geq j$.
- $(s_j, \sigma) \models f_1 \ \mathsf{U} \ f_2$ iff there exists $k \geq j$ such that $(s_k, \sigma) \models f_2$ and for all $i, j \leq i < k$, $(s_i, \sigma) \models f_1$.

Often [Bacchus and Kabanza, 1996; 1998; Baral *et al.*, 2001] planning with respect to LTL goals are formalized with the assumption that there is complete information about the initial state, and the actions are deterministic. In that case plans are finite sequences of actions. But since truth of LTL formulas are defined with respect to a reference state and a trajectory made up of an infinite sequence of states, to define the correctness of a plan (consisting of a finite sequence of actions) with respect to an initial state and an LTL goal, we need to identify a trajectory that corresponds to the initial state and the plan. This is defined as follows:

Let s be a state designated as the initial state, let $a_1, \dots, a_n$ be a sequence of deterministic actions whose effects are described by a domain description, and Φ be transition function that defines transition between states due to actions. The trajectory corresponding to s and $a_1, \dots, a_n$ is the sequence $s_0, s_1, \dots$, that satisfies the following conditions: $s = s_0$, $s_{i+1} = \Phi(a_{i+1}, s_i)$, for $0 \leq i \leq n-1$, and $s_{j+1} = s_j$, for $j \geq n$. We then say that the sequence of actions $a_1, \dots, a_n$ is a plan from the initial state s for the goal f, if $(s, \sigma) \models f$, where σ is the trajectory corresponding to s and $a_1, \dots, a_n$.

The role of LTL in specifying planning goals has been well studied and examples of that can be found in [Bacchus and Kabanza, 1996; 1998; Niyogi and Sarkar, 2000; Baral *et al.*, 2001].

2.2 Goal representation using branching time temporal logic

The use of a branching temporal logic in specifying planning goals that can not be specified using LTLs is relatively recent [Niyogi and Sarkar, 2000; Baral *et al.*, 2001; Pistore and Traverso, 2001]. The necessity of branching time operators arises when we want to specify conditions on other paths starting from the states in the main path that the agent's plan suggests. For example, a robot going from position A to position B may be required to take a path so that from any point in the path there is a charging station within two steps. Note that these two steps do not have to be in the path of the robot. This goal can not be expressed using LTLs and we need to use a branching time logic such as CTL*. We now give the syntax and semantics for CTL* [Emerson and J.Srinivasan, 1989; Emerson, 1990].

There are two kinds of formulas in CTL*: state formulas and path formulas. Normally state formulas are properties of states while path formulas are properties of paths. The syntax of state and path formulas is as follows. Let $\langle p \rangle$ denote an atomic proposition, $\langle sf \rangle$ denote state formulas, and $\langle pf \rangle$ denote path formulas.

$$\langle sf \rangle ::= \langle p \rangle \mid \langle sf \rangle \wedge \langle sf \rangle \mid \langle sf \rangle \vee \langle sf \rangle \mid \neg \langle sf \rangle \mid$$
$$\mathsf{E} \langle pf \rangle \mid \mathsf{A} \langle pf \rangle$$
$$\langle pf \rangle ::= \langle sf \rangle \mid \langle pf \rangle \vee \langle pf \rangle \mid \neg \langle pf \rangle \mid \langle pf \rangle \wedge \langle pf \rangle \mid$$
$$\langle pf \rangle \ \mathsf{U} \ \langle pf \rangle \mid \bigcirc \langle pf \rangle \mid \Diamond \langle pf \rangle \mid \Box \langle pf \rangle$$

Recall that the symbols A and E are the branching time operators meaning 'for all paths' and 'there exists a path' respectively. As the qualification 'branching time' suggests, specification in the branching time logic CTL* are evaluated with respect to the branching structure of the time. The term 'path' in the meaning of A and E refers to a path in the branching structure of time. The branching structure is specified by a transition relation R between states of the world. Intuitively, $R(s_1, s_2)$ means that the state of the world can change from s_1 to s_2 in one step. Given a transition relation R and a state s, a path in R starting from s is a sequence of states $s_0, s_1, \dots$ such that $s_0 = s$, and $R(s_i, s_{i+1})$ is true.

In [Niyogi and Sarkar, 2000; Baral *et al.*, 2001] $R(s_1, s_2)$ is tied to actions as follows. When planning in an environment where our agent is the only one that can make changes to the world, $R(s_1, s_2)$ is true if there exists an agent's action a such that $s_2 = \Phi(s_1, a)$. If there are external agents other than our agent then $R(s_1, s_2)$ is true if there exists an action (by some agent) a such that $s_2 = \Phi(s_1, a)$. We now give the formal semantics of CTL*.

Formal semantics: Semantics of CTL* formulas are defined depending on whether they are state formulas or path formulas. The truth of state formulas are defined with respect to a pair (s_j, R), where s_j is a state and R is the transition relation. In the following p denotes a propositional formula sf_is are state formulas and pf_is are path formulas.

- $(s_j, R) \models p$ iff p is true in s_j.
- $(s_j, R) \models \neg sf$ iff $(s_j, R) \not\models sf$.
- $(s_j, R) \models sf_1 \wedge sf_2$ iff $(s_j, R) \models sf_1$ and $(s_j, R) \models sf_2$.
- $(s_j, R) \models sf_1 \vee sf_2$ iff $(s_j, R) \models sf_1$ or $(s_j, R) \models sf_2$.
- $(s_j, R) \models \mathsf{E} \ pf$ iff there exists a path σ in R starting from s_j such that $(s_j, R, \sigma) \models pf$.
- $(s_j, R) \models \mathsf{A} \ pf$ iff for all paths σ in R starting from s_j we have that $(s_j, R, \sigma) \models pf$.

The truth of path formulas are defined with respect to a triplet (s, R, σ) where σ given by the sequence of states $s_0, s_1, \dots$, is a path, R is a transition relation and s is a state in σ.

- $(s_j, R, \sigma) \models sf$ iff $(s_j, R) \models sf$.
- The truth of path formulas of the form $pf_1 \ \mathsf{U} \ pf_2$, $\neg pf$, $pf_1 \wedge pf_2$, $pf_1 \vee pf_2$, $\bigcirc pf$, $\Box pf$, and $\Diamond pf$ is defined very similar to the definition of truth of LTL formulas, and R does not play a role in it.

We say a sequence of actions $a_1, \dots, a_n$ is a plan with respect to the initial state s_0 and a goal G if $(s_0, R, \sigma) \models G$, where σ is the trajectory corresponding to s_0 and $a_1, \dots, a_n$. (Note that a trajectory corresponding to s_0 and $a_1, \dots, a_n$ is a path.) Although state formulas are also path formulas, since the evaluation of state formulas do not take into account the trajectory suggested by a prospective plan, often the overall

goal of a planning problem is better expressed as a path formula which is not a state formula.

Now we can represent the goal of getting to B such that from anywhere in the path we can get to a state where p holds in at most two steps by the following path formula (which is not a state formula) in CTL*: $(p \vee \mathsf{E} \bigcirc p \vee \mathsf{E} \bigcirc \mathsf{E} \bigcirc p) \, \mathsf{U} \, at\_B$

Additional examples of the use of branching temporal logics CTL and CTL* to specify goals are given in [Bacchus and Kabanza, 1996; 1998; Niyogi and Sarkar, 2000; Baral *et al.*, 2001; Pistore and Traverso, 2001].

3 Dealing with non-deterministic actions and policies

In this section we show how to expand on the notions in the previous section so as to be able to specify goals such as goal_a from Section 1. To start with in presence of non-deterministic actions, we need to expand the notion of a plan from a simple sequence of actions to a *policy which is a mapping from states to actions*. This is necessary because in presence of non-deterministic actions an agent can not be sure during planing time what state it would be in after executing an action or a sequence of actions. Thus often there may not exist a conformant plan consisting of sequence of actions, while there may exist a policy that will achieve a goal.

Now let us reconsider our attempts to specify goal_a (which is to to try to reach DEP but avoiding LAB) using CTL*. Earlier we argued that $\mathsf{E}(\Box \neg LAB \wedge \Diamond \Box DEP)$ is not an appropriate specification as it just requires the robot to have an available path to DEP and in that sense plan_b (where the robot goes south from STORE and north from SW) would be a valid plan with respect to this goal which is not intended. The problem here is the way we characterize exist path expressed by E. *The 'exist path' condition as formally defined in the previous section does not take into account the policy the agent is following.* In this particular example, it does not really matter if there exists a path if that path is not in tune with the agent's policy; what matters is that the path that exists be in tune with the policy. To express this we introduce the operators E_π which means that there exists a path in tune with the policy π. Now if we express our goal as $\mathsf{E}_\pi(\Box \neg LAB \wedge \Diamond \Box DEP)$ plan_b will no longer be a valid plan with respect to this goal while plan_a will be. We now formally define the syntax and semantics of this extended branching time logic, which we will refer to as π-CTL*.

3.1 Syntax of π-CTL*

The syntax of state and path formulas in π-CTL* is as follows. Let $\langle p \rangle$ denote an atomic proposition, $\langle sf \rangle$ denote state formulas, and $\langle pf \rangle$ denote path formulas.

$$\langle sf \rangle ::= \langle p \rangle \mid \langle sf \rangle \wedge \langle sf \rangle \mid \langle sf \rangle \vee \langle sf \rangle \mid \neg \langle sf \rangle \mid$$
$$\mathsf{E}\langle pf \rangle \mid \mathsf{A}\langle pf \rangle \mid \mathsf{E}_\pi\langle pf \rangle \mid \mathsf{A}_\pi\langle pf \rangle$$
$$\langle pf \rangle ::= \langle sf \rangle \mid \langle pf \rangle \vee \langle pf \rangle \mid \neg\langle pf \rangle \mid \langle pf \rangle \wedge \langle pf \rangle \mid$$
$$\langle pf \rangle \, \mathsf{U} \, \langle pf \rangle \mid \bigcirc \langle pf \rangle \mid \Diamond \langle pf \rangle \mid \Box\langle pf \rangle$$

The new symbols A_π and E_π are the branching time operators meaning 'for all paths that agree with the policy that is being executed' and 'there exists a path that agrees with the policy

that is being executed' respectively. Since we now allow actions to be non-deterministic the transition function Φ is now a mapping from states and actions to a *set of states*. Now we need to consider two transition relations R and R_π, the first used for defining paths for A and E and the second used in defining paths for A_π and E_π.

Recall that $R(s, s')$ means that the state of the world can change from s to s' in one step. This could be due to an agent's action, or an exogenous action. Thus $R(s, s')$ is true if there exists an action a such that $s' \in \Phi(a, s)$. $R_\pi(s, s')$ on the other hand means that the state of the world can change from s to s' in one step by following the agent's policy π. Thus $R_\pi(s, s')$ is true if $s' \in \Phi(\pi(s), s)$. We now give the formal semantics of π-CTL*.

3.2 Formal semantics of π-CTL*:

Recall that in CTL* truth of state formulas is defined with respect to a pair (s_j, R), where s_j is a state, R is the transition relation. Here, the truth of state formulas is defined with respect to the triplet (s_j, R, R_π), where s_j and, R are as before, and R_π is the transition relation with respect to the policy π. In the following p denotes a propositional formula sf_is are state formulas and pf_is are path formulas.

- $(s_j, R, R_\pi) \models p$ iff p is true in s_j.
- $(s_j, R, R_\pi) \models \neg sf$ iff $(s_j, R, R_\pi) \not\models sf$.
- $(s_j, R, R_\pi) \models sf_1 \wedge sf_2$ iff $(s_j, R, R_\pi) \models sf_1$ and $(s_j, R, R_\pi) \models sf_2$.
- $(s_j, R, R_\pi) \models sf_1 \vee sf_2$ iff $(s_j, R, R_\pi) \models sf_1$ or $(s_j, R, R_\pi) \models sf_2$.
- $(s_j, R, R_\pi) \models \mathsf{E} \, pf$ iff there exists a path σ in R starting from s_j such that $(s_j, R, R_\pi, \sigma) \models pf$.
- $(s_j, R, R_\pi) \models \mathsf{A} \, pf$ iff for all paths σ in R starting from s_j we have that $(s_j, R, R_\pi, \sigma) \models pf$.
- $(s_j, R, R_\pi) \models \mathsf{E}_\pi \, pf$ iff there exists a path σ in R_π starting from s_j such that $(s_j, R, R_\pi, \sigma) \models pf$.
- $(s_j, R, R_\pi) \models \mathsf{A}_\pi \, pf$ iff for all paths σ in R_π starting from s_j we have that $(s_j, R, R_\pi, \sigma) \models pf$.

The truth of path formulas are now defined with respect to the quadruplet (s_j, R, R_π, σ), where $s_j, R,$ and R_π are as before and σ given by the sequence of states $s_0, s_1, \ldots,$ is a path.

- $(s_j, R, R_\pi, \sigma) \models sf$ iff $(s_j, R, R_\pi) \models sf$.
- $(s_j, R, R_\pi, \sigma) \models \neg pf$ iff $(s_j, R, R_\pi, \sigma) \not\models pf$
- $(s_j, R, R_\pi, \sigma) \models pf_1 \wedge pf_2$ iff $(s_j, R, R_\pi, \sigma) \models pf_1$ and $(s_j, R, R_\pi, \sigma) \models pf_2$.
- $(s_j, R, R_\pi, \sigma) \models pf_1 \vee pf_2$ iff $(s_j, R, R_\pi, \sigma) \models pf_1$ or $(s_j, R, R_\pi, \sigma) \models pf_2$.
- $(s_j, R, R_\pi, \sigma) \models \bigcirc pf$ iff $(s_{j+1}, R, R_\pi, \sigma) \models pf$.
- $(s_j, R, R_\pi, \sigma) \models \Box pf$ iff $(s_k, R, R_\pi, \sigma) \models pf$, for all $k \geq j$.
- $(s_j, R, R_\pi, \sigma) \models \Diamond pf$ iff $(s_k, R, R_\pi, \sigma) \models pf$, for some $k \geq j$.
- $(s_j, R, R_\pi, \sigma) \models pf_1 \, \mathsf{U} \, pf_2$ iff there exists $k \geq j$ such that $(s_k, R, R_\pi, \sigma) \models pf_2$, and for all $i, j \leq i < k$, $(s_i, R, R_\pi, \sigma) \models pf_1$.

3.3 Plans/policies for π-CTL* goals:

Now we need to define when a mapping π from states to actions is a plan with respect to a π-CTL* goal G, an initial state s_0, and a transition function Φ from states and actions to sets of states. We denote this by $(s_0, \pi, \Phi) \models G$. For this we need to define the set of trajectories that the world may go through when we start from s_0 and follow π. Note that when actions are deterministic there is exactly one such trajectory. This is not the case when actions could be non-deterministic.

Definition 2 Given an initial state s_0, a policy π, and a transition function Φ, $\sigma_{\pi, s_0, \Phi}$ is the set of all trajectories $s_0, s_1, s_2, s_3 \ldots$ such that $s_{i+1} \in \Phi(\pi(s_i), s_i)$.

Definition 3 Given an initial state s_0, a policy π, a goal G and a transition function Φ, we say π is a plan with respect to G, s_0 and Φ (denoted by $(s_0, \pi, \Phi) \models G$) if for all $\sigma \in \sigma_{\pi, s_0, \Phi}$ we have that $(s_0, R, R_\pi, \sigma) \models G$.

Unlike before, now that we are dealing with policies (instead of sequences of actions) representing the overall goal as a state formula is no longer less preferable.

3.4 The example from Section 1

Let us consider the example from Section 1 and analyze if $G_1 = \mathsf{E}_\pi(\Box \neg LAB \wedge \Diamond \Box DEP)$ is an appropriate representation of goal_a and show that plan_a is indeed a plan that achieves goal_a while plan_b does not.

Let us denote the states $\{STORE\}$, $\{SW\}$, and $\{DEP\}$ by s_0, s_1 and s_2 respectively. Let π_1 denote plan_a where we have $\pi_1(s_0) = south$, $\pi_1(s_1) = east$, and $\pi_1(s_2) = no\_op$, where no_op is an action that does not affect any state. Now $\sigma_{\pi_1, s_0, \Phi}$ is the infinite set of trajectories $s_0, (s_1)^n, s_2, s_2, \ldots$, where $n \geq 1$ (and n is a natural number) plus the trajectory $s_0, s_1, s_1, \ldots$.

Now let us explore if for all σ in $\sigma_{\pi_1, s_0, \Phi}$ we have $(s_0, R, R_{\pi_1}, \sigma) \models G_1$. By following the definitions:
$(s_0, R, R_{\pi_1}, \sigma) \models G_1$ iff $(s_0, R, R_{\pi_1}) \models G_1$ iff
there exists path σ' in R_{π_1} starting from s_0 such that $(s_0, R, R_{\pi_1}, \sigma') \models \Box \neg LAB \wedge \Diamond \Box DEP$

The last entailment is indeed true but it illustrates a flaw in our representation of goal_a, as we are only looking for a path starting from s_0. What is intended is that such path exist from all states in the trajectory.

Thus, the appropriate representation of goal_a which reflects this aspect is $G_2 = \Box \mathsf{E}_\pi(\Box \neg LAB \wedge \Diamond \Box DEP)$. We now have the following result.

Proposition 1 $(s_0, \pi_1, \Phi) \models G_2$.

Note that $(s_0, \pi_1, \Phi) \models G_2$ iff for all σ in $\sigma_{\pi_1, s_0, \Phi}$ we have $(s_0, R, R_{\pi_1}, \sigma) \models G_2$. Now even if we take σ as $s_0, s_1, s_1, \ldots$, it is still the case that $(s_0, R, R_{\pi_1}, \sigma) \models G_2$, as for all j, $(s_j, R, R_{\pi_1}, \sigma) \models G_1$ iff $(s_j, R, R_{\pi_1}) \models G_1$, and indeed $(s_j, R, R_{\pi_1}) \models G_1$ is true as there always exists a path σ' in R_{π_1} starting from any s_j such that $(s_j, R, R_{\pi_1}, \sigma') \models \Box \neg LAB \wedge \Diamond \Box DEP$.

Now let us consider π_2 (which denotes plan_b) where we have $\pi_2(s_0) = south$, $\pi_2(s_1) = north$.

Proposition 2 $(s_0, \pi_2, \Phi) \not\models G_2$.

Proof: $\sigma_{\pi_2, s_0, \Phi}$ is the set consisting of the single trajectory $\sigma_1 = s_0, s_1, s_0, s_1, \ldots$. Now it is easy to see that $(s_0, R, R_{\pi_1}, \sigma_1) \not\models G_2$ as $(s_0, R, R_{\pi_1}, \sigma_1) \not\models G_1$ as $(s_0, R, R_{\pi_1}) \not\models G_1$ as there does not exist a path σ' in R_{π_2} starting from s_0 such that $(s_0, R, R_{\pi_2}, \sigma') \models \Box \neg LAB \wedge \Diamond \Box DEP$.

4 Relation with Dal Lago et al.'s formulation

In this section we consider some of the constructs from [Dal Lago *et al.*, 2002] and show how they can be expressed using π-CTL*. Their goal language is defined as follows where p denotes propositional formulas and g denotes extended goals.

$p ::= \top \mid \bot \mid \neg p \mid p \vee p \mid p \wedge p$
$g ::= p \mid g \textbf{ And } g \mid g \textbf{ Then } g \mid g \textbf{ Fail } g \mid \textbf{ Repeat } g \mid$
$\quad \textbf{DoReach } p \mid \textbf{TryReach } p \mid \textbf{DoMaint } p \mid$
$\quad \textbf{TryMaint } p$

We now explain some of the important and novel constructs above and show how they can be expressed in π-CTL*. (In the sequel we will give the formal characterization of the above language and do a formal comparison.)

1. **TryReach p** : The intended meaning of this goal is that an agent policy that satisfies this goal must do its best to reach p. That means if the agent follows its policy then at every state reached while following this policy there is a path (which is possible by following the agent's policy but not necessarily guaranteed by the agent's policy) from that state to a state where p is true.

   This can be expressed in π-CTL* by $\Box \mathsf{E}_\pi \Diamond p$.

   If the intention is that once p is reached p must remain true after that, then in that case the specification would be $\Box \mathsf{E}_\pi \Diamond \Box p$.

   Also, if the intention is that the existence of the path be only true at the initial state s_0 then we can remove the $\Box$ in the beginning and either $\mathsf{E}_\pi \Diamond p$ or $\mathsf{E}_\pi \Diamond \Box p$ would suffice.

   The above alternatives point to a drawback of the restricted and specialized syntax of [Dal Lago *et al.*, 2002], using which one can only represent one of the above specifications; not all of them.

   Finally, for the reasons discussed earlier in the paper replacing E_π simply by E is not adequate.

2. **TryMaint p** : The intended meaning of this goal is that an agent policy that satisfies this goal must do its best to maintain p. That means if the agent follows its policy then at every state reached while following this policy there is a path (which is possible by following the agent's policy but not necessarily guaranteed by the agent's policy) from that state where p is true all through the path.

   This can be expressed in π-CTL* by $\Box \mathsf{E}_\pi \Box p$.

   If the intention is that the existence of the path be only true at the initial state s_0 then we can remove the $\Box$ in the

beginning and $\mathsf{E}_\pi \Box p$ would suffice. Note that $\mathsf{E}\Box p$ is not adequate as it just says that there exists a path where p is maintained. This path could be due to actions not dictated by the policy being followed, and if that is the case the policy being followed is not really trying to maintain p.

3. **DoReach** p : The intended meaning of this goal is that an agent policy that satisfies this goal must take the world to a state where p is true. That means if the agent follows its policy then no matter what path it takes (due to the non-deterministic effect of actions), all those paths lead to a state where p is true.

   This can be expressed in π-CTL* by $\mathsf{A}_\pi \diamond p$.

   If the intention is that once p is reached p must remain true after that, then in that case the specification would be $\mathsf{A}_\pi \diamond \Box p$. Note that both $\mathsf{A}\diamond p$ and $\mathsf{A}\diamond\Box p$ are too strong as they require that all paths (including paths not dictated by the policy being executed) lead to a state where p is true.

4. **DoMaint** p : The intended meaning of this goal is that an agent policy that satisfies this goal must take paths where p is true in all states of the path. That means if the agent follows its policy then no matter what path it takes (due to the non-deterministic effect of actions), p is true in all the states of those paths.

   This can be expressed in π-CTL* by $\mathsf{A}_\pi \Box p$. Here also, $\mathsf{A}\Box p$ is too strong a specification.

5. **Repeat** g : The intended meaning of this goal is to repeatedly achieve g. A simple representation of this in π-CTL* (and also in LTL) is $\Box\diamond g$.

6. g_1 **Fail** g_2 : The intended meaning of this goals is to achieve g_1 and when it becomes clear that g_1 is not achievable then g_2 is achieved. Let us consider a specific case of this: **TryReach** p **Fail DoReach** p. This can be expressed in π-CTL* by $\Box(\neg\mathsf{E}_\pi\diamond p \Rightarrow \mathsf{A}_\pi\diamond q)$.

5 Conclusion and future work

In this paper we considered representing goals with temporal aspects in presence of non-deterministic actions. We analyzed why temporal logics such as LTL and CTL* were thought to be not adequate to express certain kind of goals. By our analysis we discovered the source of confusion: the notion of $path$ that is tied to the branching time operators A and E. We showed that by introducing additional branching time operators A_π and E_π where the path is tied to the policy being executed we can express the goals that were thought inexpressible using temporal logic in [Dal Lago *et al.*, 2002].

We believe our approach is preferable to the approach in [Dal Lago *et al.*, 2002] where a specialized language is introduced. This is because by using our language we can still represent the goals that were traditionally represented (when actions were assumed to be deterministic) using temporal logic, and also combine such goals with the kind of goals discussed in this paper. Use of a specialized language, as in [Dal Lago *et al.*, 2002], makes this difficult if not impossible.

In terms of future work we need to consider some further generalizations. For example, in certain cases we may need to distinguish between paths solely due to actions that can be executed by an agent, and paths solely due to exogenous actions. Or we may need to consider paths that interleave agent's actions and exogenous actions in a particular way such as alternating them. Each of these may necessitate use of additional transition relations (such as R_a corresponding to agent's actions, and R_e corresponding to exogenous actions) and additional branching time operators (such as E_e, A_e, E_a, and E_e).

We also plan to further elaborate on the π-CTL* specifications of the various constructs of [Dal Lago *et al.*, 2002]. Finally, we also plan to consider alternative notions of maintainability [Nakamura *et al.*, 2000] and explore its relation with the formulations in this paper.

References

[Bacchus and Kabanza, 1996] F. Bacchus and F. Kabanza. Planning for temporally extended goals. In *AAAI 96*, pages 1215–1222, 1996.

[Bacchus and Kabanza, 1998] F. Bacchus and F. Kabanza. Planning for temporally extended goals. *Annals of Math and AI*, 22:5–27, 1998.

[Baral *et al.*, 2001] C. Baral, V. Kreinovich, and R. Trejo. Computational complexity of planning with temporal goals. In *IJCAI 2001*, 2001.

[Dal Lago *et al.*, 2002] U. Dal Lago, M. Pistore, and P. Traverso. Planning with a language for extended goals. In *AAAI'02*, pages 447–454, 2002.

[Emerson and J.Srinivasan, 1989] E. A. Emerson and J.Srinivasan. Branching time temporal logic. In J.W. de Bakker, W. P. de Roever, and G. Roezenberg, editors, *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 123–172. Springer-Verlag, Berlin, 1989.

[Emerson, 1990] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of theoretical computer science: volume B*, pages 995–1072. MIT Press, 1990.

[Manna and Pnueli, 1992] Z. Manna and A. Pnueli. *The temporal logic of reactive and concurrent systems: specification*. Springer Verlag, 1992.

[Nakamura *et al.*, 2000] M. Nakamura, C. Baral, and M. Bjareland. Maintainability: a weaker stabilizability like notion for high level control. In *Proc. of AAAI'00*, pages 62–67, 2000.

[Niyogi and Sarkar, 2000] R. Niyogi and S. Sarkar. Logical specification of goals. In *Proc. of 3rd international conference on Information Technology*, pages 77–82, 2000.

[Pistore and Traverso, 2001] M. Pistore and P. Traverso. Planning as model checking for extended goals in non-deterministic domains. In *IJCAI'01*, 2001.