Towards Strong Cyclic Planning under Partial Observability

Piergiorgio Bertoli[1], **Alessandro Cimatti**[1], **Marco Pistore**[2]

[1] ITC-Irst, via Sommarive 18, Povo, Italy, e-mail [bertoli,cimatti]@irst.itc.it
[2] DIT - University of Trento, via Sommarive 14, Povo, Italy, e-mail pistore@dit.unitn.it

Abstract

Strong Cyclic Planning aims at generating iterative plans that only allow loops so far as there is a chance to reach the goal. The problem is already significantly complex for fully observable domains; when considering partially observable domains, even providing a formal definition is far from straightforward. In this work, we provide a formal definition of Strong Cyclic Planning under Partial Observability, which makes clear how several degrees of solution are possible and equally interesting, depending on the admissible delay between achieving the goal and detecting that it has been achieved.

Introduction

Planning is nondeterministic domains has received very strong attention in the last few years, for its ability to relax some of the basic assumptions underlying classical planning (Kabanza, Barbeau, & St-Denis 1997; Weld, Anderson, & Smith 1998; Bertoli *et al.* 2001; Rintanen 1999; Bonet & Geffner 2000; Jensen, Veloso, & Bowling 2001; Cimatti *et al.* 2003). Dealing with nondeterminism is very challenging for a number of reasons: for instance, due to the uncertain initial conditions and to nondeterministic action effects, a plan is associated with multiple runs.

In *strong* planning, a solution must *guarantee* that the goal is achieved in a finite number of steps *for all runs* associated with a plan. Often, however, strong plans may not exist. It is therefore reasonable to tackle the problem of finding a *strong cyclic solution*, i.e. a conditional plan that either achieves the goal in a finite number of steps, or goes through a possibly infinite but *good* loop, where we still have the possibility of reaching the goal. Trial-and-error strategies of this form are necessary in many domains, where e.g. the interference from uncontrollable agents make an iterative behavior necessary.

The problem of strong cyclic planning has been first addressed in (Cimatti, Roveri, & Traverso 1998), under the hypothesis of full observability, and later on also tackled by (Kuter *et al.* 2005; Cimatti *et al.* 2003; Jensen, Veloso, & Bowling 2001). This paper is a preliminary step towards Strong Cyclic Planning under Partial Observability. In particular, we provide a precise characterization of the problem of strong cyclic planning under partial observability. This turns out not to be straightforward, since different interpretations are possible, and equally interesting. In particular, partial observability results in uncertainty at execution time, which may prevent or delay the detection of goal achievement. We distinguish different degrees of solutions where a plan can (i) achieve the goal and detect achievement at the same time, (ii) achieve the goal and detect achievement after some delay, and (iii) achieve the goal without ever detecting achievement.

Planning in Nondeterministic Domains

In our framework, a planning domain is a generic transition system, possibly with its own dynamics (e.g. a power plant or an aircraft). A planning domain is defined in terms of its *states*, of the *actions* it accepts, and of the possible *observations* that the domain can exhibit. Some of the states are marked as valid *initial states* for the domain. A *transition function* describes how (the execution of) an action leads from one state to possibly many different states. Finally, an *observation function* defines what observations are associated to each state of the domain.

Definition 1 (planning domain) *A* nondeterministic planning domain with partial observability *is a tuple* $\mathcal{D} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{I}, \mathcal{T}, \mathcal{X} \rangle$*, where:*

- $\mathcal{S}$ *is the set of* states.
- $\mathcal{A}$ *is the set of* actions.
- $\mathcal{O}$ *is the set of* observations.
- $\mathcal{I} \subseteq \mathcal{S}$ *is the set of* initial states*; we require* $\mathcal{I} \neq \emptyset$.
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow 2^{\mathcal{S}}$ *is the* transition function*; it associates to each current state* $s \in \mathcal{S}$ *and to each action* $a \in \mathcal{A}$ *the set* $\mathcal{T}(s, a) \subseteq \mathcal{S}$ *of next states.*
- $\mathcal{X} : \mathcal{S} \rightarrow 2^{\overline{\mathcal{O}}}$ *is the* observation function*; it associates to each state* s *the set of possible observations* $\mathcal{X}(s) \subseteq \mathcal{O}$.

We say that action a *is* executable *in state* s *if* $\mathcal{T}(s, a) \neq \emptyset$. *We require that in each state* $s \in \mathcal{S}$ *there is some executable action, and that some observation is associated to each state* $s \in \mathcal{S}$ *(i.e.,* $\mathcal{X}(s) \neq \emptyset$*).*

If $Q \subseteq \mathcal{S}$, we write $\mathcal{X}(Q)$ for $\{o \in \mathcal{O} \mid \exists s \in Q.o \in \mathcal{X}(s)\}$, and $\mathcal{X}^{-1}(o)$ for $\{s \in \mathcal{S} \mid o \in \mathcal{X}(s)\}$. This model of observation is very general: for instance, it allows modeling noisy sensing, by associating more than one observation to a given state.

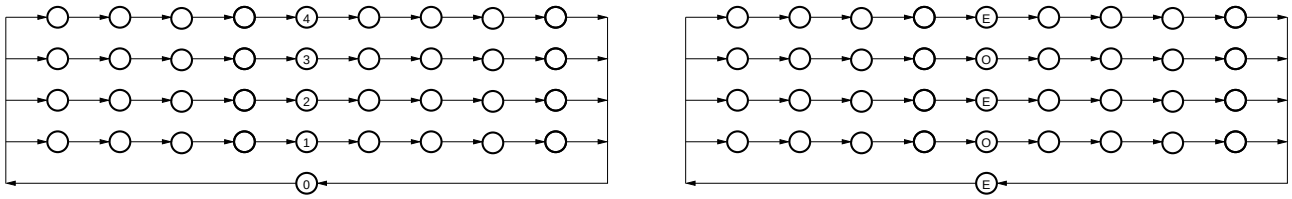

Figure 1: A simple explanatory robot domain

Example 2 *Consider the domain composed of floors from 0 to F, and rooms from 0 to $R = 2 \times F$ (depicted in Figure 1 for F=4). At the ground floor, composed of a single room, the robot can* MoveUp *with an elevator, which lands unpredictably at any floor. Then it can only* MoveLeft*, until it gets at room R, where it can* MoveDown *back to ground floor. All the rooms are indistinguishable, except for the middle room M of each floor (with $M = \lceil R/2 \rceil$), where a tag displays the current floor. The states of the domain are the pairs of the form $\langle f \ . \ r \rangle$, where $f \in 1 \dots F$ is the floor and $r \in 0 \dots R$ is the room of the robot, and $\langle 0 \ . \ M \rangle$ denotes the only location of the ground floor. The set of observations is $\{0, 1, \dots, F, \bullet\}$. The observation function is defined as follows: $\mathcal{X}(\langle f \ . \ r \rangle) = f$ iff $r = M$ (i.e., when in the room with the tag, the floor is perceived), and $\mathcal{X}(\langle f \ . \ r \rangle) = \bullet$ in the other locations (i.e. $\bullet$ represents the absence of information).*

We are interested in complex plans that may encode sequential, conditional and iterative behaviors. We model them as finite state machines, as follows.

Definition 3 (plan) *A* plan *for planning domain $\mathcal{D} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{I}, \mathcal{T}, \mathcal{X} \rangle$ is a tuple $\Pi = \langle \mathcal{C}, c_0, \alpha \rangle$, where:*

- *$\mathcal{C}$ is the set of* plan contexts*.*
- *$c_0 \in \mathcal{C}$ is the* initial context*.*
- *$\alpha : \mathcal{C} \times \mathcal{O} \to 2^{\mathcal{A} \times \mathcal{C}}$ is the* action relation*; it associates to a plan context c and an observation o a set of pairs (a, c'), where a is an action to be executed and c' is a new plan context.*

The *context* is the internal state of the plan. Actions to be executed depend on the context, and on the observation. Once an action is executed, the context is updated depending on the observation and on the action that has been carried out. Plans implement nondeterministic behaviors, i.e. different actions and new contexts may be taken (at different times during execution) when the plan is in the same context and receives the same observation. We call a plan deterministic when $\alpha(c, o)$ contains at most one pair (a, c') for a given action a. The execution of a plan terminates when a context c is reached and an observation o is done for which $\alpha(c, o)$ is empty.

Example 4 *With the plan $\Pi^{\langle f \ . \ M+f \rangle}_{\text{ONE}}$, the robot takes the elevator, moves to the middle room where it looks at the floor tag, and stops after f steps, where f is the number displayed on the tag. The plan has $R + 1$ contexts, of which S_{Init} is initial and T_0 is final; α is defined as follows:*

c	o	a	c'	
S_{Init}	0	MoveUp	S_0	
S_i	$\bullet$	MoveLeft	S_{i+1}	$i = 0 \dots M-1$
S_M	f	MoveLeft	T_{f-1}	
T_i	$\bullet$	MoveLeft	T_{i-1}	$i = 1..F$

In plan $\Pi^{\langle f \ . \ f \rangle}_{\text{ONE}}$, initiating at context S_{Init} and terminating at context S_M, the robot proceeds to traverse the first M rooms, and then stops.

c	o	a	c'	
S_{Init}	0	MoveUp	S_0	
S_i	$\bullet$	MoveLeft	S_{i+1}	$i = 0 \dots M-1$

In the plan $\Pi^{\langle 2 \ . \ M+2 \rangle}_{\text{CYCLE}}$, the robot iterates until it sees the floor 2 tag, then proceeds for two steps, and then stops at context T_0. α for $\Pi^{\langle 2 \ . \ M+2 \rangle}_{\text{CYCLE}}$ is defined as follows:

c	o	a	c'	
S_{Init}	0	MoveUp	S_0	
S_i	$\bullet$	MoveLeft	S_{i+1}	$i \neq M$
S_M	$f = 2$	MoveLeft	T_1	
S_M	$f \neq 2$	MoveLeft	S_{M+1}	
S_R	$\bullet$	MoveDown	S_{Init}	
T_1	$\bullet$	MoveLeft	T_0	

In a similar way, it is possible to define a plan $\Pi^{\langle 2 \ . \ M-2 \rangle}_{\text{CYCLE}}$, where the robot iterates until it detects it is in floor 2, and in that case it stops, and a plan Π^{*}_{CYCLE} such that the robot simply loops forever in the domain.

Defining Strong Cyclic Planning

We are interested in finding plans that achieve the goal despite uncertainty in the initial condition and nondeterministic action effects. The notion of *strong* planning defined in (Cimatti *et al.* 2003) requires that the condition is achieved in a finite number of steps. Unfortunately, in many situations this requirement may be too strong. Therefore, we are willing to accept iterative plans, which implement trial-and-error courses of actions, and might result in possibly infinite behaviors. With *strong cyclic planning*, we require that any loop still has a chance to reach the goal, and we avoid bad loops, from which the goal will be unreachable.

In the fully observable case, after the execution of an action, we are always able to decide if the goal has been reached; therefore, strong cyclic planning only has to fight

against nondeterminism (Cimatti *et al.* 2003). In the partially observable case, we face the additional difficulty that it may be impossible to detect whether the goal has been reached.

In order to illustrate this point, we start by considering some problems in our example domain that can be solved by strong planning. If our goal is to reach and detect that the robot is at $\langle f \ . \ R - f \rangle$, plan $\Pi_{\text{ONE}}^{\langle f \ . \ M + f \rangle}$ is a solution: the uncertainty about the position of the robot after the ride in the elevator disappears after looking at the sensors, so that at the end of every possible plan execution, we are sure that the desired condition is achieved. Consider now the goal of reaching one of the locations $\langle f \ . \ f \rangle$. With the plan $\Pi_{\text{ONE}}^{\langle f \ . \ f \rangle}$ execution, the robot will traverse the required state. However, the plan does not stop the robot in one of the required positions – in fact, this goal would be unsolvable. Thus, at the end of the plan execution, all we can guarantee is that "the goal has been achieved sometimes during the execution". This example highlights the fundamental difference between a plan that only achieves a certain goal condition, and a plan that, in addition, guarantees that the executor will know when the goal is achieved (and will stop accordingly). The first interpretation is the standard one used in strong planning under partial observability: a solution plan is associated with final belief states that are entirely contained in the goal. The second interpretation is new: a solution plan guarantees that, for each of the states in any final belief states, a goal state was previously visited. Accordingly, we distinguish between goal achievement *with immediate detection*, and goal achievement *with delayed detection*. Obviously, the requirement of immediate achievement detection is harder to satisfy.

Let us now turn to the general case of strong cyclic planning. Consider the goal $\langle 2 \ . \ M + 2 \rangle$. It is easy to see that it admits no strong solution in either interpretation. However, if we admit the possibility of infinite executions, the problem of goal achievement with immediate detection is solvable, and plan $\Pi_{\text{CYCLE}}^{\langle 2 \ . \ M + 2 \rangle}$ is a strong cyclic solution. Either the goal is reached (and immediately detected), or the system loops through states with a possibility of reaching the goal, going back to ground floor and restarting, hoping that the elevator stops at floor 2. The goal $\langle 2 \ . \ M - 2 \rangle$ is unsolvable under the immediate detection hypothesis: in fact, even if the robot is at the right floor (i.e. floor 2), it has to leave the goal location in order to read the floor tag situated in room 4 in order to be sure that the goal location has been traversed in its past history. The problem therefore admits a strong cyclic solution with delayed achievement detection – one example is the plan $\Pi_{\text{CYCLE}}^{\langle 2 \ . \ M - 2 \rangle}$.

Consider now a variation of the domain, where sensors are not precise and associate the same information to adjacent floors. Then, due to the robot being unable to localize its floor, it is not possible to achieve the goal $\langle 2 \ . \ M + 2 \rangle$ and detect it (immediately or later). However a cyclic solution exists if goal detection is given up completely, by looping in the domain with plan Π_{CYCLE}^{*}.

In the rest of this section, we formalize the notions outlined above. We describe the execution of a plan over a

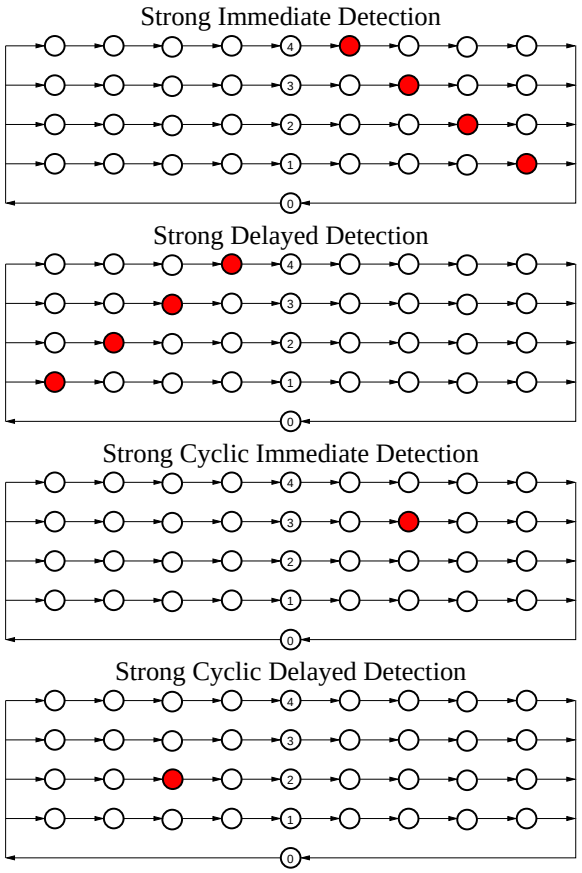

Figure 2: Different goals for different problem classes.

domain, in terms of transitions between configurations that describe the state of the domain and of the plan.

Definition 5 (configuration) *A* configuration *for domain* $\mathcal{D} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{I}, \mathcal{T}, \mathcal{X} \rangle$ *and plan* $\Pi = \langle \mathcal{C}, c_0, \alpha \rangle$ *is a tuple* (s, o, c) *such that*

- $s \in \mathcal{S}$,
- $o \in \mathcal{X}(s)$,
- $c \in \mathcal{C}$.

Configuration (s, o, c) *may* evolve into *configuration* (s, o, c), *written* $(s, o, c) \rightarrow (s', o', c')$, *if there is some action* a *such that:*

- $(a, c') \in \alpha(c, o)$,
- $s' \in \mathcal{T}(s, a)$,
- $o' \in \mathcal{X}(s')$.

Configuration (s, o, c) *is* initial *if* $s \in \mathcal{I}$ *and* $c = c_0$.
The reachable configurations *for domain* $\mathcal{D}$, *plan* Π, *and goal* $\mathcal{G}$, *are defined by the following inductive rules:*

- *if* (s, o, c) *is initial, then it is reachable;*
- *if* (s, o, c) *is reachable and* $(s, o, c) \rightarrow (s', o', c')$, *then* (s', o', c') *is also reachable.*

Configuration (s, o, c) *is* terminal *iff it is reachable and* $\alpha(c, o) = \emptyset$.

Intuitively, a configuration is a snapshot of the domain controlled by plan execution. Observations are included in configurations in order to take into account that more than one observation may correspond to the same state.

Definition 6 (path) *A finite [infinite]* path P *is a finite [infinite] sequence of configurations* $\gamma_0, \gamma_1, \ldots$ *such that* γ_0 *is initial and* $\gamma_i \rightarrow \gamma_{i+1}$ *for all* i*. Prefixes of a path and path extensions are defined in the usual way.*
We say that path P *contains a state* s *iff there exists a configuration* $\gamma_i = (s_i, o_i, c_i) \in P$ *s.t.* $s_i = s$*.*
We denote with $\textsc{Paths}(\mathcal{D}, \Pi)$ *be the set of that paths for* $\mathcal{D}$ *and* Π *that either are infinite or end with a terminal configuration.*

Different paths may be *observationally equivalent*, i.e. undistinguishable for an external observer:

Definition 7 (equivalent paths) *Two paths* $P = (s_0, o_0, c_0), (s_1, o_1, c_1), \ldots$ *and* $P' = (s'_0, o'_0, c'_0), (s'_1, o'_1, c'_1), \ldots$ *are said to be equivalent, denoted with* $P \equiv P'$*, iff for every* i*,* $o_i = o'_i$*.*

We are interested in plans that guarantee the applicability of actions to be executed for each reachable configuration.

Definition 8 (applicable plan) *Plan* Π *is* applicable *on domain* $\mathcal{D}$ *iff for each reachable configuration* (s, o, c)*, if* $(a, c') \in \alpha(c, o)$*, then* $\mathcal{T}(s, a) \neq \emptyset$*.*

We are now ready to formally characterize strong and strong cyclic solutions under partial observability:

Definition 9 (strong and strong cyclic solution) *Let plan* Π *be applicable in domain* $\mathcal{D}$*.* Π *is a* strong solution *to* $\mathcal{G}$ *for*

- achievement with immediate detection (SID) *iff* $\textsc{Paths}(\mathcal{D}, \Pi)$ *contains only finite paths, and* $s \in \mathcal{G}$ *holds for each terminal configuration* (s, o, c)*;*
- achievement with delayed detection (SDD) *iff* $\textsc{Paths}(\mathcal{D}, \Pi)$ *contains only finite paths, and each (finite) path in* $\textsc{Paths}(\mathcal{D}, \Pi)$ *contains some state* $s \in \mathcal{G}$*.*

Π *is a* strong cyclic solution *to* $\mathcal{G}$ *for*

- achievement with immediate detection (SCID) *iff* $s \in \mathcal{G}$ *holds for each terminal configuration* (s, o, c)*, and each prefix of an infinite path in* $\textsc{Paths}(\mathcal{D}, \Pi)$ *is observationally equivalent to some prefix of a finite path in* $\textsc{Paths}(\mathcal{D}, \Pi)$*.*
- achievement with delayed detection (SCDD) *iff each finite path in* $\textsc{Paths}(\mathcal{D}, \Pi)$ *contains some state* $s \in \mathcal{G}$*, and each prefix of an infinite path in* $\textsc{Paths}(\mathcal{D}, \Pi)$ *is observationally equivalent to some prefix of a finite path in* $\textsc{Paths}(\mathcal{D}, \Pi)$*.*
- achievement without detection (SCND) *iff each (finite and infinite) path in* $\textsc{Paths}(\mathcal{D}, \Pi)$ *contains some state* $s \in \mathcal{G}$*.*

In the acyclic case, the existence of a solution implies that of a solution with delayed detection: when the execution terminates (which always occurs in a finite number of steps) we are guaranteed that a state in $\mathcal{G}$ has been encountered.

Also notice that SCND could be defined differently, by requiring that each prefix of an infinite path is equivalent to a prefix of a (finite of infinite) path containing a goal state. However this would make it impossible to monitor the satisfaction of the goal at runtime - something granted by each execution in the current definition, and, for what concerns SCDD and SCID, correspondent to plan termination. Thus we prefer to provide the current definition, where every infinite execution is known to satisfy the goal.

Conclusions and Future Work

In this paper we argue that defining the problem of strong cyclic planning under partial observability is not straightforward, and we identify several different (but equally interesting) definitions of the problem. In fact, only (Iocchi, Nardi, & Rosati 2004) tackles strong cyclic planning in the context of partial observability, but it provides just an informal statement of the problem which completely overlooks the issue of goal detection. Of course, the issue of (effectively) identifying the solutions for the given problems is open, and our next step will consist in defining planning algorithms to this purpose, and evaluating their performance. Symbolic techniques such as those used in (Cimatti *et al.* 2003) seem promising candidates in this respect.

References

Bertoli, P.; Cimatti, A.; Roveri, M.; and Traverso, P. 2001. Planning in Nondeterministic Domains under Partial Observability via Symbolic Model Checking. In *Proc. IJCAI'01*.

Bonet, B., and Geffner, H. 2000. Planning with Incomplete Information as Heuristic Search in Belief Space. In *Proceedings of AIPS'00*, 52–61. AAAI Press.

Cimatti, A.; Pistore, M.; Roveri, M.; and Traverso, P. 2003. Weak, Strong, and Strong Cyclic Planning via Symbolic Model Checking. *Artificial Intelligence* 147(1-2):35–84.

Cimatti, A.; Roveri, M.; and Traverso, P. 1998. Automatic OBDD-based Generation of Universal Plans in Non-Deterministic Domains. In *Proc. AAAI'98*.

Iocchi, L.; Nardi, D.; and Rosati, M. 2004. Strong Cyclic Planning with Incomplete Information and Sensing. In *Proc. of 4th International Workshop on Planning and Scheduling for Space*.

Jensen, R. M.; Veloso, M. M.; and Bowling, M. H. 2001. OBDD-Based Optimistic and Strong Cyclic Adversarial Planning. In *Proc. of ECP'01*.

Kabanza, F.; Barbeau, M.; and St-Denis, R. 1997. Planning Control Rules for Reactive Agents. *Artificial Intelligence* 95(1):67–113.

Kuter, U.; Nau, D.; Pistore, M.; and Traverso, P. 2005. A Hierarchical Task Network Planner based Symbolic Model Checking. In *Proceedings of ICAPS'05*, 300–309.

Rintanen, J. 1999. Constructing Conditional Plans by a Theorem-Prover. *Journal of Artificial Intellegence Research* 10:323–352.

Weld, D.; Anderson, C.; and Smith, D. 1998. Extending Graphplan to Handle Uncertainty and Sensing Actions. In *Proc. AAAI'98*, 897–904.