6.825 Techniques in Artificial Intelligence

Planning Miscellany

- SATPlan
- Conditional Planning

SATPLAN

- One approach: Extract SAT problem from planning graph
- Another approach: Make a sentence for depth n, that has a satisfying assignment iff a plan exists at depth n
  - Variables:
    - Every proposition at every even depth index: $clean_0$, $garb_2$
    - Every action at every odd depth index: $cook_1$

Constructing SATPLAN sentence

- Initial sentence (clauses): $garb_0$, $clean_0$, $quiet_0$, $\neg present_0$, $\neg dinner_0$
- Goal (at depth 4): $\neg garb_4$, $present_4$, $dinner_4$
- $Action_t \rightarrow (Pre_{t-1} \wedge Eff_{t+1})$ [in clause form]
  - $Cook_1 \rightarrow (clean_0 \wedge dinner_2)$
- Explanatory Frame Axioms: For every state change, say what could have caused it
  - $garb_1 \wedge \neg garb_3 \rightarrow (dolly_2 \vee carry_2)$ [in clause form]
- Conflict exclusion: For all conflicting actions a and b at depth t, add $\neg a_t \vee \neg b_t$
  - One's precondition is inconsistent with the other's effect

SATPLAN

- There are many preprocessing steps possible to reduce the size of the SAT problem
- We can use insight of where sentence came from to, for example, choose the order of the variables in DPLL [pick action variables first, they cause conflicts as soon as possible].
- Recently, new methods that are closer to first order have become more popular

Planning Assumptions

learning {
- Assumed complete and correct model of world dynamics

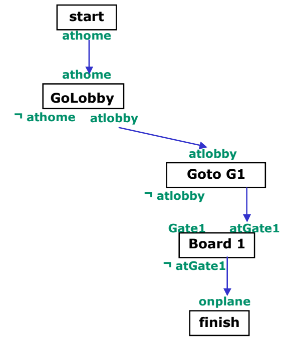
conditional planning / replanning {
- Assumed know initial state
- Assumed world is deterministic

- These assumptions hold in domains such as scheduling machines in factories but not in many other domains.

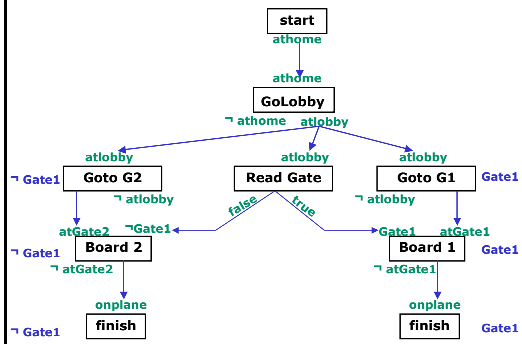
Conditional Planning Example

Action	Preconditions	Effects
ReadGate	AtLobby	KnowWhether(Gate1)
BoardPlane1	Gate1, AtGate1	OnPlane, ¬AtGate1
BoardPlane2	¬Gate1, AtGate2	OnPlane, ¬AtGate2
GotoLobby	AtHome	AtLobby, ¬AtHome
GotoGate1	AtLobby	AtGate1, ¬AtLobby
GotoGate2	AtLobby	AtGate2, ¬AtLobby

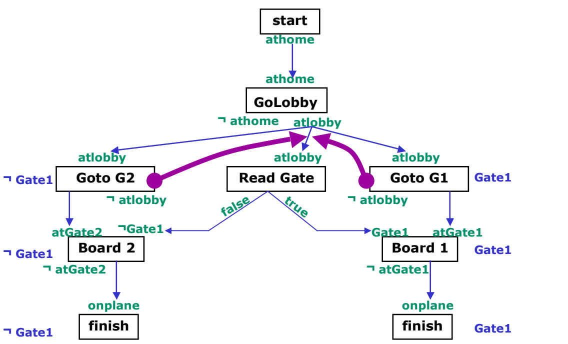
Partial Order Conditional Plan



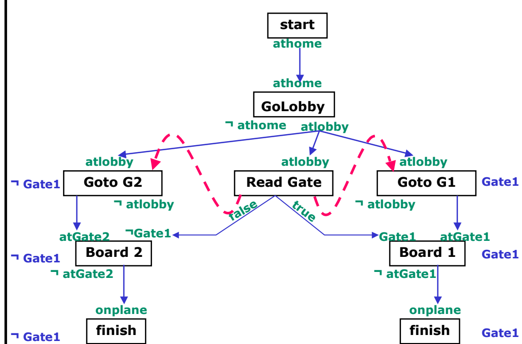
Partial Order Conditional Plan



Partial Order Conditional Plan



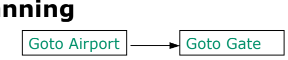
Partial Order Conditional Plan



Conditional Planning

- POP with these new ways of fixing threats and satisfying preconditions increases the branching factor in the planning search and makes POP completely impractical
- People are working on conditional planning versions of GraphPlan and SatPlan
- Instead of constructing conditional plans ahead of time, just plan as necessary when you have the information.

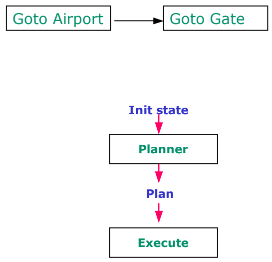
Replanning



- One place where replanning can help is to fill in the steps in a very high-level plan

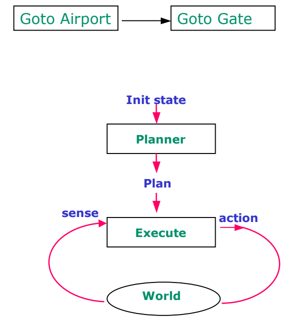
Replanning

- One place where replanning can help is to fill in the steps in a very high-level plan
- Another is to overcome execution errors



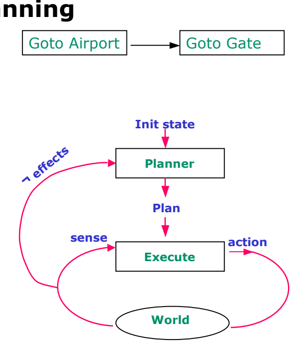
Replanning

- One place where replanning can help is to fill in the steps in a very high-level plan
- Another is to overcome execution errors



Replanning

- One place where replanning can help is to fill in the steps in a very high-level plan
- Another is to overcome execution errors



Replanning Cycle

Universal Plan

Assume
- Offline computation is cheap
- Space is plentiful
- Online computation is expensive

- Plan for every possible initial state
- Store: initial state $\rightarrow$ first step

- World is completely observable

Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Execute Highest True Kernel (rectangle including lower left corner and some upper right corner)

Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Execute Highest True Kernel (rectangle including lower left corner and some upper right corner)

Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Execute Highest True Kernel (rectangle including lower left corner and some upper right corner)

Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Execute Highest True Kernel (rectangle including lower left corner and some upper right corner)

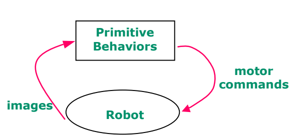
Triangle Tables
Fikes & Nilsson

	Init				
Pre(A_1)	Sells(HW, Drill)	**Go HW** *Eff(A_1)*			
Pre(A_2)		At HW	**Buy Drill** *Eff(A_2)*		
Pre(A_3)		At HW		**Go SM** *Eff(A_3)*	
Pre(A_4)	Sells(SM, Bananas)			At SM	**Buy Ban** *Eff(A_4)*
Goal Conds			Have Drill		Have Bananas

Execute Highest True Kernel (rectangle including lower left corner and some upper right corner)

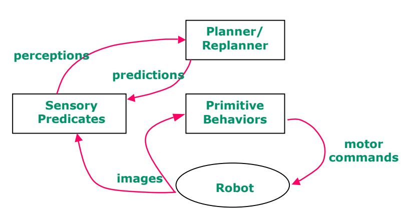
Hybrid Architectures

- Reactive lower level
- Deliberative higher level



Hybrid Architectures

- Reactive lower level
- Deliberative higher level



Hybrid Architectures

- Reactive lower level
- Deliberative higher level