CSE 513T Theory of Artificial Intelligence & Machine Learning April 02, 2015

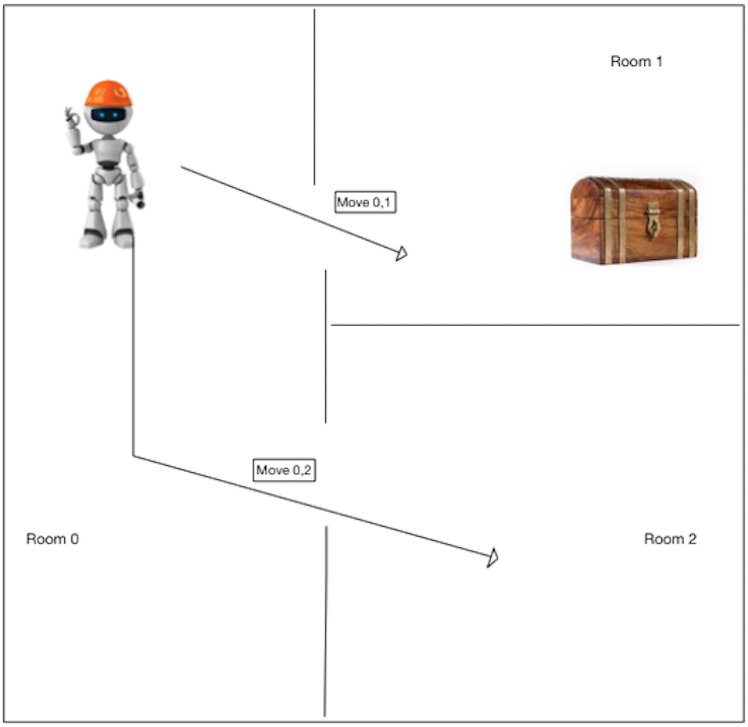
Lecture 22

Lecturer: Brendan Juba *Scribe: Jinxuan Zhao*

1 Today's Topics

1. Classical (STRIPS) Planning
2. Logical Encoding of Action and Change
3. Planing-as-satisfiability

2 Planning Problem:



Graph 1:Robot in room 0, box in room 1.

1. Basics:
   - Planning $\longleftrightarrow$ How to produce intelligent behavior. (Application: Robotics)
   - Agent autonomously select a series of actions in persuit of some goal.
   - To understand the effect of actions and their relationship, we need a model of environment (A representation of it), in which the robot agent operates.
   - STRIPS domains: Particulary simple class of actions, environments used by 'shakey' the robot at SRI in the 1960s.

2. What is a STRIPS domain ?

   - Environment $\longleftrightarrow$ List of attributes.
     Example: Attributes encoding the robot's location.
     $at_{r,i} \equiv$ robot (r) is in room i.
     $at_{b,i} \equiv$ box (b) is in room i
     $holds_{r,b} \equiv$ robot (r) is holding the box (b).
     In the state graph (Graph 1) above:
     $at_{r,0} = 1$, $at_{r,1} = 0$, $at_{r,2} = 0$
     $at_{b,0} = 0$, $at_{b,1} = 1$, $at_{b,2} = 0$
     $holds_{r,b} = 0$

3 About actions and changes:

1. Basics concepts:

   - Action: "move to room i", "Pick a box", "Put down the box"
   - Move: Defined by their effects:
     $move_{0,1}$ has the effect of setting $at_{r,0} = 0$ and $at_{r,1} = 1$.
     Write above as literals that will be satisfied: $(\neg\ at_{r,0},\ at_{r,1})$ for $move_{0,1}$.
   - Generally, effects are a set of literals that become satisfied.
   - There are "Pre-conditions" that has to be satisfied for actions to be executable. In general, if we have $move_{i,j}$, then it requires $at_{r,i}$ to be 1, if we have $PickUp_{b,i}$, it requires $at_{r,i}$ and $at_{b,i}$ to be 1.

     In general, pre-conditions are also a set of literals that must be satisfied.
     Example : For $PickUp_b$, we need $(at_{r,0} \wedge at_{b,0}) \bigvee (at_{r,1} \wedge at_{b,1}) \bigvee (at_{r,2} \wedge at_{b,2})$

     Note: Making this condition so definable requires the action to be subscripted with the location, otherwise we'd need a precondition like: *Is the robot in room 0 and box in room 0 or ...*

   - Conditional effects:
     If $holds_{r,b}$ and robot moves,, then the box should move with it. We encode the moving of the box as a conditional effect of $move_{i,j}$. If $hold_{r,b}$ is satisfied then $move_{i,j}$ sets $at_{b,i}$ to 0 and $at_{b,j}$ to 1.

     In general: conditional effects of an action are given by an effect literal together with a sets of precondition *(for this effect, not the action).*

     Note: Even if the conditional effect preconditions are not satisfied, the agent may still execute the action.
   - Goal: A STRIPS goal is given by a conjunction. (AND of literals over the state attributes)

     A plan in a STRIPS domain is a sequence of actions. (S.T the pre-condition of each action are satisfied-needs knowledge of state)

     The plan achieves the goal if the goal fomula is satisfied on the state of environment after the final action of the plan.

- A STRIPS planning problem consists of:
  1: A STRIPS domain: complete description of attributes, actions, and effects.
  2: An initial state: setting of the state attributes.
  3: A STRIPS goal.

- Solution: A plan that transforms the initial state to one in which the goal condition is satisfied in the STRPIS domain.

2. Example:
   In our robot domain, start from the initial condition state:

   - Goal to achieve: $(at_{b,2} \wedge \neg at_{r,2})$
     $at_{b,2}$: Put the box in room 2. $\neg at_{r,2}$: The robot needs to be outside of room 2.

   - A solution would be the following:
     1: Move to room 1 $\longleftrightarrow move_{0,1}$
     2: Pick up the box $\longleftrightarrow PickUp_{b,1}$
     3: Move to room 0 $\longleftrightarrow move_{1,0}$
     4: Move to room 2 $\longleftrightarrow move_{0,2}$
     5: Put down the box $\longleftrightarrow putDown_2$
     6: Move to room 0 $\longleftrightarrow move_{2,0}$

Step	$at_{r,0}$	$at_{r,1}$	$at_{r,2}$	$at_{b,0}$	$at_{b,1}$	$at_{b,2}$	$hold_{r,b}$
0	1	0	0	0	1	0	0
1	0	1	0	0	1	0	0
2	0	1	0	0	1	0	1
3	1	0	0	1	0	0	1
4	0	0	1	0	0	1	1
5	0	0	1	0	0	1	0
6	1	0	0	0	0	1	0

Table 1: The table for corresponding states of environment

4 Logical encoding of Actions and Change:

- Our objective represent the entire planing problem as a CNF, use SAT-Solver to find plans.
- In particular, we'll also encode the choice of actions as a propositional attributes so that given a satisfying assignment, we can "read off" the plan directly.
- Small problems:
  1: The settings the state and action attributes need to change during the execution of the plan. (Attributes that changes value are called "fluents")
  2: Fix include a copy of each fluen attribute for each step of the plan, including a copy of X, X_t for $t = 0, 1, 2.....T$. (T is the time horizon.)
- We need to fix a time horizon T.
  Without one we do know that:
  n fluents $\Rightarrow$ 2^n possible states $\Rightarrow$ A plan that has more than 2^n steps must have visited a state more than once so we could omit the segment of the plan on this "loop" in state space, since we only care about the final state.

- Typically, we wouldn't be interested in such long plans, so we just consider the **bounded** versions of STRIPS planing given a planing problem together with a time horizon T, find a plan of at most T steps.

$n * T$ variables	fluents
$t = 0$	$X_{1,0}$,$X_{2,0}$.....$X_{n,0}$
1	$X_{1,1}$,$X_{2,1}$.....$X_{n,1}$
...	...
T	$X_{1,T}$,$X_{2,T}$.....$X_{n,T}$

Table 2:Action attributes for each action α , attribute $\alpha_{\alpha,t} \equiv$ action α taken at step t.

- We need to assume the existence of/add a *wait* action that preserves the state of the environment. A shorter plan can be extended to one that is precisly T steps by adding *wait* actions.
(Could optimize length of plan by trying to solve a sequence of instances of bounded horizon planing by using binary search on T.)

- Encoding an action with Pre-conditions. (As multiple clauses)
Effect has pre-conditions $q_1....q_s$.
Literal e_j, $j = 1....r$
$[q_{1,t} \wedge ...q_{s,t} \wedge \alpha_{\alpha,t}] \Rightarrow e_{1,t+1}$
$[q_{1,t} \wedge ...q_{s,t} \wedge \alpha_{\alpha,t}] \Rightarrow e_{r,t+1}$
Action's precondition:
$p_1....p_k$
$[\alpha_{\alpha,t}] \Rightarrow p_{1,t}$
$[\alpha_{\alpha,t}] \Rightarrow p_{k,t}$

- **What we still need...**
1: Rules describing what doesn't change as a result of an action in a step.

2: Rules specifying that a unique action is taken at each set. (This is straightforward, add clauses. $(\vee_{\alpha \in A}, \alpha_{\alpha,t}) \wedge \bigwedge_{\alpha \neq \alpha' \in A} \neg(\alpha_{\alpha,t} \wedge \alpha_{\alpha',t}), t = 0...T-1$.

5 Next Class:

- The frame problem: Elegant solution using negation-as-failure..
$[l_t \wedge not(\neg l_{t+1})] ==\Rightarrow l_{t+1}$, but in our encoding as a CNF, we'll have to reset to something messier.