CogSysI Lecture 6: AI Planning: Current Approaches

Intelligent Agents

WS 2005/2006

Part I: Acting Goal-Oriented

AI Planning: Current Approaches
Domain-independent planning in deterministic domains

- 70ies: STRIPS and deductive planning
- 80ies: partial order planning (e.g. UCPOP)
- 90ies: extensions of STRIPS language (PDDL) and new, efficient algorithms – Graphplan, SATPlan
Graphplan: Basic Ideas

- Make search for a plan more efficient by first constructing a **planning graph** from which the valid plan can be extracted.

- A planning graph is a directed, levelled graph, that is, nodes can be partitioned into disjoint sets $L_1, L_2, \ldots, L_n$ such that the edges connect only nodes in adjacent levels.

- Two kinds of nodes:
  - Starting with level 0, nodes at even levels represent propositions true at time $t_i$.
  - Nodes at odd levels represent possible actions at time $t$. 
Graphplan: Basic Ideas cont.

Four kinds of edges
- Precondition edges: from propositions to actions
- Add edges: from actions to propositions
- Del edges: from actions to propositions
- No-op edges: from propositions to propositions
move(?r ?f ?t)
PRE: (at ?r ?f), (?f ≠ ?t), (has-fuel ?r)
ADD: (at ?r ?t)
DEL: (at ?r ?f), (has-fuel ?r)

unload(?r ?p ?c)
PRE: (at ?r ?p) (in ?c ?r)
ADD: (at ?c ?p)
DEL: (in ?c ?p)

load((?r ?p ?c)
PRE: (at ?r ?p) (at ?c ?r)
ADD: (in ?c ?p)
DEL: (at ?c ?p)

Example Problem:
Initial State: { (at R L), (at A L), (at B L), (has-fuel R) }
Goal: { (at A P), (at B P) }
Construct Planning Graph

- Start with initial state
- Introduce all propositions as nodes on level $S_0$
- Construct the next levels:
  - Find all actions whose preconditions are contained in level $S_i$ and introduce them as nodes in level $A_i$
  - Introduce all propositions of the ADD and DEL list in level $S_{i+1}$
  - Copy all propositions from level $S_i$ to level $S_{i+1}$
Rocket Planning Graph
Remarks on Planning Graphs

can be constructed in polynomial time since planning is NP-hard (PSPACE complete), plan extraction is of exponential effort!

based on propositional logic may be hard to transform a problem given in PDDL into propositional form! (all possible instantiation of actions, which instances are legal?)

represent a partially-ordered plan (independent actions are given on the same level and can be performed in arbitrary order) actions are independent, if one does not delete a precondition or an add-effect of the other

to facilitate plan extraction, mutual exclusive nodes can be identified
Mutex Relations

Two actions at one level are mutex, if
- either one deletes a precondition or add-effect of the other (interference)
- their preconditions are mutex

Two propositions are mutex, if
actions adding them are mutex.

Mutex Relations of the Rocket Problem:
- (move L P R) and (load A L R) are mutex because the precondition for “load” is deleted by “move”

Mutex Relations are a heuristic:
- Not all mutex relations can be found by this rules
- More complex mutex relations (between triples etc.) are not checked (too expensive)
Extraction of a Valid Plan

by backwards-search (goal regression)

- Level-by-level to exploit the mutex constraints
- Given a set of goals at level $i$, find a set of actions (including no-ops) at the preceding level which have these goals as add-effects
- The preconditions of these actions form the subgoals for the next regression step
- if a goal set at a level is unsolvable, backtrack and select different actions
- continue until success or prove that the original goals are not solvable at this level ($\Rightarrow$ expand planning graph to a next level containing all goals)
Termination on unsolvable problems is tricky, but the Graphplan algorithm is sound and complete for STRIPS-style operators where only a finite set of propositions can be generated.
Planning in Real-World Domains

- Incomplete Information
  - Conformant planning: Create plans that work for all cases
  - Conditional planning: sense world during execution and decide which branch of the plan to follow

- Incorrect Information
  - Execution monitoring: check for unsatisfied preconditions
  - Re-planning

- Continuous planning: create new goals during acting in real time

- Multiagent planning
Including Knowledge

Using knowledge about the structure of the domain

- Hierarchical Planning (decomposition rules)
  cf. problem solving with AND-OR trees
- Domain axioms
- Domain specific search strategies

→ larger plans become feasible (necessary for many real world problems, e.g. Mars Mobile)
Alternative to knowledge engineering: Learning of planning strategies!