Reminder: Last Lecture

What is Planning

Planning and Search

Logical Approach

STRIPS (Classical) Approach

Forward Search

Regression Search

Planning: Outline of lectures

Logical representation

- Situation calculus

STRIPS representation

- States and Actions
- Examples

Simple search methods

- Forward search
- Backward search
- Heuristic search

Planning: Outline (cont)

Partial Order Planning

- Search with grounded values
- Search with variables
- Heuristics

Planning with Graphs

- Planning graphs
- Heuristics
- “Graphplan” methods

Regression planning ("Backwards search")

Basic Idea:

- Start with the goal condition
- Select an action that achieves some element in goal condition
- Find a state that permits action and gives goal condition
- Apply recursively to the state found
- Use search method of choice

Finding regression state of given state S:

- Select a literal in S, say it is L (and L is either p or –p)
- Find action A that achieves L
  - has p in add effects if L=p
  - has p in del effects if L= -p
- Regressed state is (s \ L ∪ pre(A))

Example:

Initial state:

- loc(A,table)
- loc(B,C)
- loc(C,table)
- clear(A)
- clear(B)

Goal condition:

- loc(B,A)
- loc(A,C)

About regression planning

More focused than forward search

- Only examines actions that in some way relate to the goal

Blind regression is impractical

- Blind search is invariably exponential in length of plan

Heuristics more complex than for forward search

- But, some promising techniques based on heuristic regression

Heuristic state space search

Blind search does not work in planning

- Branching factor is way too large
- Without heuristics, even small planning problems are unsolvable

Basic ideas for heuristics in state space search for planning

- Simplify preconditions – ignore some of them
- Simplify effects – ignore some of them

Heuristics from ignoring preconditions

Extreme version: Ignore all preconditions

- Assume every action can be applied in any state
- Find a plan from given state to a goal state
  - Easier problem to solve, but additions and effects still make it nontrivial
- Length of plan is heuristic evaluation of s

Variations

- Assume effects of actions are independent
  - Length of plan then becomes number of literals in goal different from s
- Assume actions of have no delete effects
  - Finding plan then becomes very easy

Heuristics from ignoring effects

Basic idea

- Assume actions have no delete effects
- Solve the simplified planning problem from state s to goal state
- Length of plan is heuristic evaluation of state s

Implementation

- Solving a planning problem to get heuristics
- But, problem is much simpler and easier to solve

Separating subgoals

Basic idea

- Find a plan for each literal in goal condition
- Combine the plans to generate a complete plan

Problem!

- Initial state: loc(B,table), loc(A,table), loc(C,A), clear(B), clear(C)
- Goal: loc(A,B), loc(B,C)
- Called the Sussman anamoly

Partial Order Planning

Basic Idea

- Want to be able to solve subgoals in parallel
- Often unnecessary to decide subgoal ordering too early
  - General notion: "Least commitment planning"
- POP: Work with plans where actions have not necessarily yet been ordered

Key Issue

- How to reason about state if plan is not fully ordered?

Partial Order Planning

Extra actions for initial and goal states

- Start: All actions must come after this action.
  Effects of action are the full initial state specification
- End: All actions must come before this action.
  Precondition of action is the goal condition

Partially ordered plan as candidate solution

- Could enforce full ordering, but it is not necessary
- If any ordering of partial order is a legal plan,
  then partial order is okay

Partial Order Planning and Search

Candidate partial order plans

- Set of actions in plan (including start and end)
- Ordering constraints
- Causal links to establish preconditions
- Open preconditions

Consistent partial order plan

- No loops in ordering constraints
- No actions violating causal links (threats)
  - Action C threatens causal link for p between A and B if C deletes p and can occur between A and B

Complete plan

- Consistent and no open preconditions

Partial Order Planning and Search

Basic idea behind search

- Nodes in search are candidate partial plans (not states)
- Search expansion defined by selecting an open precondition and generating all possible resolutions

Specific method – given candidate plan **P**

- Select an open precondition p for some action A
- For each action B, either in plan or that can be added to plan, and achieves p, generate all consistent candidate plans that have causal link from B to A for p:
  - For any C that can contradict p, add anew candidate plan for each legal ordering of C before B or after A, as long as overall ordering conditions are consistent.

Heuristics for Partial Order Planning

Plan evaluation:

- Simple idea: Count open preconditions
- Better idea: Estimate complexity of completion with plangraph

Open precondition selection:

- Simple idea: Select precondition with fewest resolution options ("cheapest first")

Plan graphs

Objective

- Get better information about states/plans
  - Simpler methods do not account for interaction between actions and states
  - Example: Sussman anomaly

Basic Idea

- Build a small and cheap structure that can still track interactions
- If it is too simple, cannot track interactions
- If it is too complex, becomes too expensive to calculate and store
- Solution: Limit interaction tracking to "mutual exclusion" (mutex)

Building a plangraph

Initialization

- Start with initial state (or state being evaluated)

Repeat as needed

- Add an action level
  - Action added if preconditions in previous state level and not mutex
  - "No op" actions added for each literal in previous state level
  - Two actions are mutex if any preconditions are mutex
  - Two actions are mutex if their effects/preconditions are inconsistent
- Add a state level
  - Add literal if there is an action that adds literal (includes "no ops")
  - Two contradictory literals are always mutex (p and –p)
  - Two literals are mutex if all action pairs to add them are mutex

Stop when level is the same as previous level

- Or, when solution has been found, if search is interleaved

About mutexes

Action mutexes:

- Two actions are mutex if effects/preconditions are incompatible
- Two actions are mutex in a given level if their preconditions have any pairwise mutexes
  - This can change between levels

State mutexes:

- Two literals are mutex if they are contradictory
- Two literals are mutex in a given level if no non-mutex actions can achieve both at same time
  - This can change between levels

Mutex properties:

- If a mutex disappears, it does not reappear in later levels

Graphplan

Graphplan

- Method to do planning, using plangraphs
- Revolutionized field of planning

Basic idea

- Add layers until all goal conditions appear without mutexes
- Search for a set of actions that form a legal plan
  - This is where the search problem appears
- If no solution is found, add a layer and repeat
- If adding layer changes nothing and no solution is found, then problem is unsolvable

Various uses for plangraphs

Structure for planning

- Graphplan and many more planners

Heuristics evaluation and pruning

- Can use plangraphs to see what states are unreachable
- Can use plangraphs to evaluate partial plans and states
  - Find depth of layer where goal conditions first all appear without mutex
  - Sum up depths of layers for each conditions
  - etc.

Planning in the “Real World”

Why does STRIPS not suffice?

- <u>Time</u> impacts planning and especially concurrent plans
- <u>Resources</u> are difficult to handle in STRIPS
  - Unary resources, such as equipment (can be done)
  - Multiple resources, such as equipment (hard, but not impossible)
  - Numerical resources, such as fuel (very hard and expensive)
  - etc.
- <u>Numerical calculations</u> are impossible in STRIPS
- <u>Complex conditions</u> and <u>state constraints</u> are very hard in STRIPS
- <u>Uncertainty</u> and <u>sensing</u> is very hard to do in STRIP

But, can still use strips in real world, if we are clever…