CSE 3401: Intro to AI & Logic Prog
Planning as Heuristic Search

- Readings: Russell & Norvig 3rd edition Chapter 10 (in 2nd edition, Sections 11.1, 11.2, and 11.4)

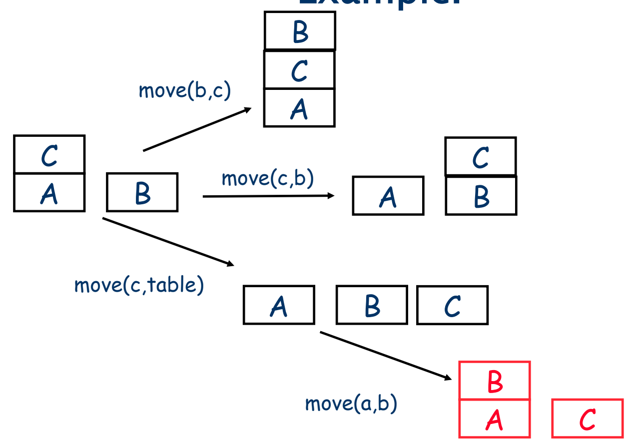
Planning as a Search Problem

- Given a CW-KB representing the initial state, a set of STRIPS or ADL (Action Description Language) operators, and a goal condition we want to achieve (specified either as a conjunction of facts, or as a formula)
  - The planning problem is determine a sequence of actions that when applied to the initial CW-KB yield an updated CW-KB which satisfies the goal.

Planning As Search

- This can be treated as a search problem.
  - The initial CW-KB is the initial state.
  - The actions are operators mapping a state (a CW-KB) to a new state (an updated CW-KB).
  - The goal is satisfied by any state (CW-KB) that satisfies the goal.

Example.



Problems

- Search tree is generally quite large
  - randomly reconfiguring 9 blocks takes thousands of CPU seconds.
- The representation suggests some structure. Each action only affects a small set of facts, actions depend on each other via their preconditions.
- Planning algorithms are designed to take advantage of the special nature of the representation.

Planning

- We will look at 1 technique
- Relaxed Plan heuristics used with heuristic search.

Reachability Analysis.

- The idea is to consider what happens if we ignore the delete lists of actions.
- This is yields a “relaxed problem” that can produce a useful heuristic estimate.

Reachability Analysis

- In the relaxed problem actions add new facts, but never delete facts.
- Then we can do reachability analysis, which is much simpler than searching for a solution.

Reachability

- We start with the initial state S_0.
- We alternate between state and action layers.
- We find all actions whose preconditions are contained in S_0. These actions comprise the first action layer A_0.
- The next state layer consists of all of S_0 as well as the adds of all of the actions in A_0.
- In general
  - A_i is the set of actions whose preconditions are contained in S_i.
  - S_{i+1} is S_i union the add lists of all of the actions in A_i.

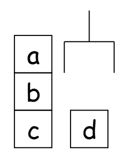
STRIPS Blocks World Operators.

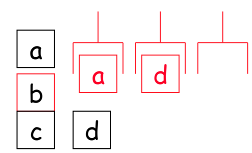
- pickup(X)
  Pre: {clear(X), ontable(X), handempty}
  Add: {holding(X)}
  Del: {clear(X), ontable(X), handempty}
- putdown(X)
  Pre: {holding(X)}
  Add: {clear(X), ontable(X), handempty}
  Del: {holding(X)}

STRIPS Blocks World Operators.

- unstack(X,Y)
  Pre: {clear(X), on(X,Y), handempty}
  Add: {holding(X), clear(Y)}
  Del: {clear(X), on(X,Y), handempty}
- stack(X,Y)
  Pre: {holding(X),clear(Y)}
  Add: {on(X,Y), handempty, clear(X)}
  Del: {holding(X),clear(Y)}

Example





on(a,b),
on(b,c),
ontable(c),
ontable(d),
clear(a),
clear(d),
handempty

S_0

unstack(a,b)
pickup(d)

A_0

on(a,b),
on(b,c),
ontable(c),
ontable(d),
clear(a),
handempty,
clear(d),
holding(a),
clear(b),
holding(d)

S_1

this is not a state!

Example

on(a,b),
on(b,c),
ontable(c),
ontable(d),
clear(a),
clear(d),
handempty,
holding(a),
clear(b),
holding(d)

S_1

putdown(a),
putdown(d),
stack(a,b),
stack(a,a),
stack(d,b),
stack(d,a),
pickup(d),
…
unstack(b,c)
…

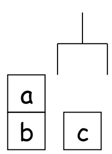
A_1

Reachabilty

- We continue until the goal G is contained in the state layer, or until the state layer no longer changes.
- Intuitively, the actions at level A_i are the actions that could be executed at the i-th step of some plan, and the facts in level S_i are the facts that could be made true after some i-1 step plan.
- Some of the actions/facts have this property. But not all!

Reachability

and on(c,b) needs 4 actions



on(a,b),
on(b,c),
ontable(c),
ontable(b),
clear(a),
clear(c),
handempty

S_0

unstack(a,b)
pickup(c)

A_0

on(a,b),
on(b,c),
ontable(c),
ontable(b),
clear(a),
clear(c),
handempty,
holding(a),
clear(b),
holding(c)

S_1

stack(c,b)
…

A_1

…
on(c,b),
…

but stack(c,b) cannot be executed after one step

Heuristics from Reachability Analysis

Grow the levels until the goal is contained in the final state level S[K].

- If the state level stops changing and the goal is not present. The goal is unachievable. (The goal is a set of positive facts, and in STRIPS all preconditions are positive facts).

- Now do the following

Heuristics from Reachability Analysis

CountActions(G,S_K):

/* Compute the number of actions contained in a relaxed plan achieving the goal. */

- Split G into facts in S_{K-1} and elements in S_K only. These sets are the previously achieved and just achieved parts of G.
- Find a minimal set of actions A whose add-effects cover the just achieved part of G. (The set contains no redundant actions, but it might not be the minimum sized set.)
- Replace the just achieved part of G with the preconditions of A, call this updated G, NewG.
- Now return CountAction(NewG,S_{K-1}) + number of actions needed to cover the just achieved part of G.

Example

legend: [pre]act[add]

$S_0 = \{f_1, f_2, f_3\}$

$A_0 = \{[f_1]a_1[f_4], [f_2]a_2[f_5]\}$

$S_1 = \{f_1,f_2,f_3,f_4,f_5\}$

$A_1 = \{[f_2,f_4,f_5]a_3[f_6]\}$

$S_2 = \{f_1,f_2,f_3,f_4,f_5,f_6\}$

$G = \{f_6,f_5, f_1\}$

We split G into G_P and G_N:

CountActs(G,S_2)

$G_P = \{f_5, f_1\}$ //already in S1

$G_N = \{f_6\}$ //New in S2

$A = \{a_3\}$ //adds all in G_N

//the new goal: $G_P \cup Pre(A)$

$G_1 = \{f_5,f_1,f_2,f_4\}$

Return

1 + CountActs(G_1,S_1)

Example

Now, we are at level S1

$S_0 = \{f_1, f_2, f_3\}$

$A_0 = \{[f_1]a_1[f_4], [f_2]a_2[f_5]\}$

$S_1 = \{f_1,f_2,f_3,f_4,f_5\}$

$A_1 = \{[f_2,f_4,f_5]a_3[f_6]\}$

$S_2 = \{f_1,f_2,f_3,f_4,f_5,f_6\}$

$G_1 = \{f_5,f_1,f_2,f_4\}$

We split G1 into G_P and G_N:

CountActs(G_1,S_1)

$G_P = \{f_1,f_2\}$ //already in S0

$G_N = \{f_4,f_5\}$ //New in S1

$A = \{a_1,a_2\}$ //adds all in G_N

//the new goal: $G_P \cup Pre(A)$

$G_2 = \{f_1,f_2\}$

Return

2 + CountActs(G_2,S_0)

= 2 + 0

So, in total CountActs(G,S2)=1+2=3

Using the Heuristic

1. To use CountActions as a heuristic, we build a layered structure from a state S that reaches the goal.
2. Then we CountActions to see how many actions are required in a relaxed plan.
3. We use this count as our heuristic estimate of the distance of S to the goal.
4. This heuristic tends to work better as a best-first search, i.e., when the cost of getting to the current state is ignored.

Admissibility

- An optimal length plan in the relaxed problem (actions have no deletes) will be a lower bound on the optimal length of a plan in the real problem.
- However, CountActions **does NOT compute** the length of the optimal relaxed plan.
- The choice of which *action set* to use to achieve G_P ("just achieved part of G") is not necessarily optimal.
- In fact it is NP-Hard to compute the optimal length plan even in the relaxed plan space.
- So CountActions will not be admissible.

Empirically

- However, empirically refinements of CountActions performs very well on a number of sample planning domains.

CWA

- "Classical Planning". No incomplete or uncertain knowledge.
- Use the "Closed World Assumption" in our knowledge representation and reasoning.
  - The Knowledge base used to represent a state of the world is a list of positive ground atomic facts.
  - CWA is the assumption that
    - a) if a ground atomic fact is not in our list of "known" facts, its negation must be true.
    - b) the constants mentioned in KB are all the domain objects.

State of the Art in Planning

- There are annual AI planning systems competitions (e.g. IPC)
- The Planning Domain Definition Language (PDDL) is a standard in the area; has sublanguages with varying expressiveness
- Several state of the art (classical) planning systems perform well on large real world problems

Current Research in Planning

- STRIPS/classical planning makes very strong assumptions: complete information, deterministic actions, static single-agent world
- Much recent work in planning addresses more general forms of planning that avoid such assumptions
- In such cases, a solution may be a branching plan (branching on observations/action outcomes), finite state automaton, or a policy
- Hierarchical planning/abstraction is useful to address large real world problems