1 handout: slides
State-space Planning
Types of Problems

- actions serial or parallel
- actions unit time or varying
- actions unit cost or varying
- minimize makespan, cost, combination, or multi-objective
- just logical fluents or metric quantities (e.g., resources) too
- off-line or on-line planning
- world controlled or predictable or unpredictable dynamics
- ‘single agent’ or other agents modifying state
- actions deterministic or stochastic
- states fully, partially, or not observable
- initial state known or unknown
- single goal state or set
- goals of achievement or maintenance
- action space discrete or continuous
- state space discrete or continuous

plan, conditional plan, policy
Frame Problems

representational: how to represent what doesn’t change
inferential: how to compute new state quickly
qualification: how to represent preconditions
Operator schema:

**Parameters:** Move(block, src, dest)

**Preconditions:** On(block, src), Clear(block), Clear(dest)

**Delete list:** On(block, src) Clear(dest)

**Add list:** On(block, dest) Clear(src)

Assume everything else is static. Closed world assumption.

Invented for Shakey (SRI).
Blocks World

Move block src dest
pre: On(block, src) Clear(block) Clear(dest)
preneg:
del: On(block, src) Clear(dest)
add: On(block, dest) Clear(src)

ToTable block src
pre: On(block, src) Clear(block)
preneg:
del: On(block, src)
add: OnTable(block) Clear(src)

FromTable block dest
pre: OnTable(block) Clear(block) Clear(dest)
preneg:
del: OnTable(block) Clear(dest)
add: On(block, dest)
Break

- asst 7
- asst 8
- project proposals
Initial: At(Home), Sells(HWS, Drill), Sells(SM, Milk), Sells(SM, Bananas)

**Go (here, there)**
- Pre: At(here)
- Post: At(there), ¬ At(here)

**Buy(store, x)**
- Pre: At(store), Sells(store, x)
- Post: Have(x)

Goal: At(Home), Have(Drill), Have(Milk), Have(Bananas)
Progression

Initial state: initial state
Branch on all applicable actions
Applicable: preconditions hold
Effects: delete deletes, then add adds
Goal reached when all goal atoms are true.
Heuristics

Planning

Heuristics
- Simple Heuristics
- Computing $H_1$
- Cake World
- EOLQs
Simple Heuristics

$h(n) = 0$

number of unachieved goals
reachability ('don’t delete'): $H_1$
Computing $H_1$

$t ← -1$
$Q ← I$

until all goals are true or $Q$ is empty,

$Q' ← \emptyset$

foreach $l ∈ Q$,

foreach $a$ that has $l$ as a precondition,

if all of $a$’s preconditions are now true,

foreach effect $e$ of $a$,

if $e$ is not already true,

record that $e$ became true at $t + 1$

add it to $Q'$

$t ← t + 1$
$Q ← Q'$

Then $\sum$ or $\max$ over goal.
Initial: Have(Cake)

**Eat:** Pre: Have(Cake)  
Post: Eaten(Cake), \( \neg \) Have(Cake)

**Bake:** Pre: \( \neg \)Have(Cake)  
Post: Have(Cake)

Goal: Have(Cake), Eaten(Cake)
What question didn’t you get to ask today?
What’s still confusing?
What would you like to hear more about?

Please write down your most pressing question about AI and put it in the box on your way out.

Thanks!