Planning

Chapter 10
Planning Difficulties

- Irrelevant Actions (buying 1 book by ISBN)
  Do we check every possible ISBN?

- Heuristic Function (buying 4 books)
  Value of states: How much better is $\text{Have}(A) \land \text{Have}(B)$ than $\text{Have}(B)$

- Problem Decomposition
  Decomposable problems decrease state size immensely

- Ad-Hoc representation
  Can we create a representation that will work for many problems?
Planning Language Representation (PDDL)

- **States**: use propositional literals:
  
  \[
  Poor,
  \]
  
  \[
  At(Plane_1, Sidney) \land At(Plane_2, Melbourne)
  \]
  
  Closed world assumption: unless specified as true, things are false

- **Init**: Initial state of system
  
  Specified as conjunction of literals: \( Poor \land Happy \)

- **Goals**: a partially specified state
  
  \( Rich \land Famous \land Miserable \) satisfies \( Rich \land Famous \)

- **Actions**: Specified in function-like syntax Ex:
  
  \[
  Action(Fly(p, from, to),
  \]
  
  \[
  Precond : At(p, from) \land Plane(p) \land Airport(from) \land Airport(to)
  \]
  
  \[
  Effect : \neg At(p, from) \land At(p, to))
  \]

  Positive effects grouped in ‘add’ list

  Negative (\( \neg \)) effects are grouped in a ‘delete’ list

  \[
  Result(s, a) = (s - Del(a)) \cup Add(a))
  \]
PDDL Cargo World

- PDDL description of a cargo problem involving loading and unloading cargo and flying it from place to place.

\[
\begin{align*}
\text{Init} & : (\text{At}(C_1, SFO) \land \text{At}(C_2, JFK) \land \text{At}(P_1, SFO) \land \text{At}(P_2, JFK) \\
& \land \text{Cargo}(C_1) \land \text{Cargo}(C_2) \land \text{Plane}(P_1) \land \text{Plane}(P_2) \\
& \land \text{Airport}(JFK) \land \text{Airport}(SFO)) \\
\text{Goal} & : (\text{At}(C_1, JFK) \land \text{At}(C_2, SFO))
\end{align*}
\]

\[\text{Action}(\text{Load}(c, p, a),)\]
\[\begin{align*}
\text{PRECOND} & : \text{At}(c, a) \land \text{At}(p, a) \land \text{Cargo}(c) \land \text{Plane}(p) \land \text{Airport}(a) \\
\text{EFFECT} & : \neg \text{At}(c, a) \land \text{In}(c, p))
\end{align*}\]

\[\text{Action}(\text{Unload}(c, p, a),)\]
\[\begin{align*}
\text{PRECOND} & : \text{In}(c, p) \land \text{At}(p, a) \land \text{Cargo}(c) \land \text{Plane}(p) \land \text{Airport}(a) \\
\text{EFFECT} & : \text{At}(c, a) \land \neg \text{In}(c, p))
\end{align*}\]

\[\text{Action}(\text{Fly}(p, \text{from}, \text{to}),)\]
\[\begin{align*}
\text{PRECOND} & : \text{At}(p, \text{from}) \land \text{Plane}(p) \land \text{Airport}(\text{from}) \land \text{Airport}(\text{to}) \\
\text{EFFECT} & : \neg \text{At}(p, \text{from}) \land \text{At}(p, \text{to}))
\end{align*}\]

(Q) What is the initial state, goal state, and actions?

(Q) Make a plan for this world. A first action could be \text{Load}(C_1, P_1, SFO).
PDDL Blocks World

- PDDL description of the Blocks world, in which a robot tries to stack blocks in specified piles.

\[
\text{Init}(\text{On}(A, \text{Table}) \land \text{On}(B, \text{Table}) \land \text{On}(C, A) \\
\quad \land \text{Block}(A) \land \text{Block}(B) \land \text{Block}(C) \land \text{Clear}(B) \land \text{Clear}(C))
\]

\[
\text{Goal}(\text{On}(A, B) \land \text{On}(B, C))
\]

\[
\text{Action}(\text{Move}(b, x, y), \\
\quad \text{Precond: } \text{On}(b, x) \land \text{Clear}(b) \land \text{Clear}(y) \land \text{Block}(b) \land \text{Block}(y) \land \\
\quad (b \neq x) \land (b \neq y) \land (x \neq y), \\
\quad \text{Effect: } \text{On}(b, y) \land \text{Clear}(x) \land \neg \text{On}(b, x) \land \neg \text{Clear}(y))
\]

\[
\text{Action}(\text{MoveToTable}(b, x), \\
\quad \text{Precond: } \text{On}(b, x) \land \text{Clear}(b) \land \text{Block}(b) \land (b \neq x), \\
\quad \text{Effect: } \text{On}(b, \text{Table}) \land \text{Clear}(x) \land \neg \text{On}(b, x))
\]

\[
\text{On}(b, x) \text{ means } b \text{ is on } x
\]

(Q) Draw the initial and goal states.

(Q) Write a plan that is a solution to the problem above.
Forward state-space search

- Initial state of search is initial state of planning problem.
- Applicable actions are all those that with valid preconditions.
- Goal test checks whether state satisfies goal.
- Step cost is usually 1.
- Drawback: State space can be huge!
  - example: 10 airports, 20 cargos, 5 planes
  - example: Blocks world with 100 blocks.

(Q) Trace forward search tree for blocks world problem.
Backward state-space search

- Initial state of search is goal state.
- Applicable actions are ‘Relevant’ ones
  Relevant = achieves a conjunct of the goal state.
  \(At(C_1, B)\) as achieved by \(Unload(C_1, p, B)\)
- Consistent actions don’t undo any desired literals (parts of the goal state)
- Given a goal state \(G\) and Apply predecessors as follows:
  Any positive effects of \(A\) that appear in \(G\) are deleted.
  Each precondition literal of \(A\) is added, unless it already appears.

(Q) Trace backward search tree for blocks world.
Heuristics for state-space search

- PDDL was specifically designed to facilitate certain heuristics

- $A^*$ review (4.1)
  \[ f(n) = g(n) + h(n) \]
  \[ f(n) = \text{estimated cost to reach solution going through } n \]
  \[ g(n) = \text{cost to reach node } n \]
  \[ h(n) = \text{estimated cost to reach solution from } n \]

- Admissible and consistent heuristic
  \[ h(n) \text{ never overestimates the true cost} \]
  \[ h(n) \leq c(n, a, n') + h(n') \]

- Relax Problem
  Remove all preconditions from actions.
  Empty delete list

- Decompose into subproblems
  Subgoal Independence
Heuristics

(Q) Give an admissible heuristic for Blocks world.

(Q) Give an admissible heuristic for Cargo world.