Software Agents

Problem Set V: PDDL and General Heuristics

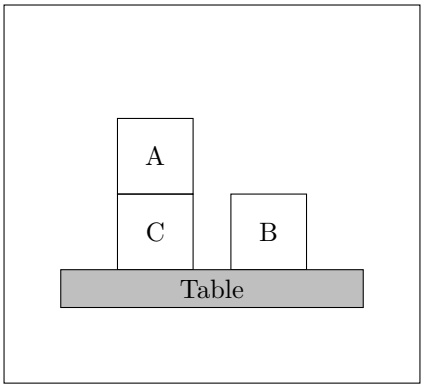

Figure 1: A blocks-world problem.

In blocks-world, the agent's aim is to stack the blocks in one tower with A on B and B on C. The actions available to the agent are: (pickup ?block ?fromBlock), (dropon ?heldBlock ?onToBlock), (pickupfromtable ?block), and (dropontable block).

and the facts: (on?b1 ?b2), (clear ?b), (ontable ?b), (holding ?b), and (handempty).

1. There are several important classes of domain-independent heuristics. Recall the critical-path based heuristics from Lectures:

- What is the h^m family of heuristics?
- Compute $h^1(s_0)$ for this blocks-world problem.
- Compute the 1-planning-graph for this blocks-world problem.

2. Implement your STRIPS model in PDDL. Use Metric-FF to test your model http://fai.cs.uni-saarland.de/hoffmann/metric-ff.html this solver is available on the department machines at /home/subjects/482/local/project/ff

A PDDL implementation is split between two files: a domain file (sometimes an "operator" file) and a problem file (sometimes a "fact" file).

The example TSP of Australia from Nir's lectures is implemented in PDDL overleaf.

See http://www.hakank.org/pddl/ for more examples.

```
(define (domain tsp)
   (:requirements :typing)
   (:types node)
   ;; Define the facts in the problem
   ;; "?" denotes a variable, "-" a type
   (:predicates (move ?from ?to - node)
                (at ?pos - node)
                (connected ?start ?end - node)
                (visited ?end - node))
   ;; Define the action(s)
   (:action move
             :parameters (?start ?end - node)
             :precondition (and (at ?start)
                                (connected ?start ?end))
             :effect  (and (at ?end)
                           (visited ?end)
                           (not (at ?start)))))
```

Figure 2: tsp-domain.pddl

```
(define (problem tsp-01)
   (:domain tsp)
   (:objects Sydney Adelade Brisbane Perth Darwin - node)
   ;; Define the initial situation
   (:init (connected Sydney Brisbane)
          (connected Brisbane Sydney)
          (connected Adelade Sydney)
          (connected Sydney Adelade)
          (connected Adelade Perth)
          (connected Perth Adelade)
          (connected Adelade Darwin)
          (connected Darwin Adelade)
          (at Sydney))
   (:goal
       (and (at Sydney)
            (visited Sydney)
            (visited Adelade)
            (visited Brisbane)
            (visited Perth)
            (visited Darwin))))
```

Figure 3: tsp-problem.pddl