

COMP3620/6320 – Planning Tutorial

Consider the following simple version of the Dock Worker Robot (DWR) domain. There is a map of *locations* that are connected pairwise. Each location can store any number of *boxes*. All boxes at a location lie on the floor instead of being stacked up on top of each other. A *robot* can carry at most one box at a time and can *move* between any two locations. In addition to robot movement, two more actions are defined. A *load* action places a box on a robot's platform, assuming that both the robot and the box are at the same location and that the robot's platform is empty. An *unload* action takes a box from a robot's platform and places it at the current location of the robot. In this simple application, loading and unloading boxes do not require the presence of a hoist.

Consider a problem instance with two locations LOC-0 and LOC-1, two boxes BOX-0 and BOX-1, and one robot ROBOT-0. Initially, both the robot and the two boxes are at location LOC-0. The goal is to have the two boxes at location LOC-1 and the robot at location LOC-0.

1. Write (part of) a STRIPS representation of the domain and the problem. For the domain representation, write the types, the predicates, and the operators.
2. Enumerate all ground (fully instantiated) actions that are generated for the problem at hand. For a few actions, show the preconditions and the effects.
3. Starting from the initial state of the problem at hand, build a planning graph with three fact levels and two action levels. (Do not explicitly compute all mutexes.)
4. Is this number of levels enough to extract a correct solution?
5. Is this number of levels enough to extract a relaxed solution? Here relaxation is obtained by ignoring all delete effects of actions.
6. Extract a relaxed solution from the planning graph.
7. Write down a correct solution for this problem.
8. Compare the relaxed solution with the correct solution. Look at the length and the actual actions included in each type of solution.
9. How can a relaxed plan be used in a search algorithm?
10. Show a few examples of fact mutexes that are permanent (i.e., they do not depend on the level of the planning graph).