Rodrigo Ipince
6.034 Project 2 – Preliminary Report
4/24/08

Exploring Mars

State and Action Representations

The state representation that we suggest is precisely the one hinted at in `rover.scm`. It consists of the action that was taken to get to that state, the current location, time, battery units left, tasks left to do, and tasks that have been completed. This information should be enough to encode the starting state and any type of goal state we want to have for each mission.

The actions used will also not be too complicated. Actions such as `go`, `drill`, `charge`, and `wait` are definitely needed, since these are the four processes that the rover can actually do. The question is then whether we would like to add actions that combine these processes in some way or another. Making higher level actions that use the wait process seems like a good idea, which is why we will implement actions such as: wait for a certain time, wait before moving, wait before communicating, and wait before charging. Other combinations do not seem necessary since the processes they would involve are relatively atomic.

Search Method and Heuristic

The best search method in terms of finding a minimum cost path fast seems to be A* with a good heuristic, which is why it is the search method that we propose to use.

A good heuristic must be able to estimate the state-distance to the goal from the given state. Thus, it should consider the sequence of drill locations that needs to be visited, the order, and the time/battery cost that each path would incur, not only in terms of time spent and battery units, but also in terms of unused time. What we mean by that is that a path with requires the rover to run out of battery during the night should be more 'expensive' than one that does not waste any time.

Coming up with a good heuristic certainly requires a deep understanding of the problem, which has not been developed yet. A first attempt at a heuristic would be to consider all drill locations that still need to be visited, pick a visiting order, calculate the Manhattan distances between each one (and between the rover's current location and the first one), pick the one that yields the lowest total cost and return that cost as the heuristic value. This heuristic disregards battery/daylight constraints, but theoretically should do a good job at finding the minimum cost path fast. This heuristic is admissible and seems to be consistent. Lastly, the usage of this heuristic highly depends on the number of drilling locations that the mission will have. If this is too high, then computing the heuristic estimate will take too long and might not be worth it.

Initial Problem Plan

The initial problem in `rover.scm` actually has no feasible solution. First, the map does not even include the initial rover location, (0 0), but is only a rectangle whose lower corner is (1 1) and upper corner is (3 3). Thus, the rover can never get to (1 1), which is the first drill location, since neither the (0 1) or (1 0) locations exist. Second, the rover starts with 10 units of initial battery power. Since each movement takes 5 units of battery power and there is no recharging (as stated in the project description), then the rover will never be able to make it back to (0 0) even if an actual path to the drill locations existed.

If we change the map to be a rectangle with lower corner (0 0) and upper corner (3 3), and let the rover start with 50 units of battery power instead, then the A* search with the proposed heuristic would yield the simple solution: move from (0 0) to (1 0), then from (1 0) to (1 1), drill, then move from (1 1) to (2 1), from (2 1) to (2 2), drill, and return to (0 0) in the same way. The whole plan would take 40 hours and cost 40 units of battery power. After adding the communication and battery charge constraints, the plan will definitely change and become more expensive.