EECS 391 (Spring 2012) Programming Assignment 3 (Max Points: 100).

Assigned Thursday March 1, due midnight Tuesday March 20. Turn in your code using blackboard. Please comment your code extensively so we can understand it, and use sensible variable names. If two people worked on a submission, please include both your names and IDs in a README.

In this assignment you will write a forward state space planner to solve a resource collection scenario in the SimpleRTS game.

1. Problem Setup

The scenarios you will solve are built around the "rc_3m5t.map" map in SimpleRTS and the "midasConfig" configuration file. In this map, there is a townhall, a peasant, three goldmines and five forests. Assume the peasant can only move between these locations. When the peasant is next to a goldmine, it can execute a HarvestGold operation. This requires the peasant to be carrying nothing and the goldmine to have some gold. If successful, it removes 50 gold from the goldmine and results in the peasant carrying 50 gold. The three goldmines in this map have capacities 100 (nearest to townhall), 500 and 5000 (farthest from townhall) respectively. When the peasant is next to a forest, it can execute a HarvestWood operation. This requires the peasant to be carrying nothing and the forest to have some wood. If successful, it removes 20 wood from the forest and results in the peasant carrying 20 wood. The five forests in this map each contain 400 wood. Finally, when the peasant is next to the townhall, it can perform a Deposit[Gold/Wood] operation. This requires the peasant to be carrying something. If successful, it results in the peasant being emptyhanded, and adds to the total quantity of gold or wood available by the amount carried by the peasant.

Note that this description goes beyond the STRIPS language in that we are describing numeric quantities. For this assignment, you should ignore this aspect and write a planner to handle this scenario assuming a "STRIPS-like" semantics according to the description above. Your code should consist of two parts: a planner and a plan execution agent (PEA). The planner will take as input the action specification above, the starting state and the goal and output a plan. The PEA will read in the plan and execute it in SimpleRTS.

Implement a forward state space planner using the A* algorithm that finds least cost plans to achieve a given goal. Here cost is time taken by the action sequence when executed. Most actions take unit time, but note that for some actions, such as compound moves, you will only be able to estimate the cost---that is fine for this assignment. Make sure to design good heuristics for the planner to guide it towards good actions---note that the branching factor of this (simple) search space is quite large once you instantiate all the ground operators! Once a plan is found, write it out to a plain text file as a list of actions, one per line, along with any parameters. Your PEA can then execute this plan in SimpleRTS.

In order to execute the found plans, you will have to translate the plan actions into SimpleRTS actions. Since you will be planning at a fairly high level, you may need to write some code to automatically choose target objects if needed so actions can execute properly. You are welcome to do this in a heuristic manner. If at some point the plan read in by the PEA is not executable in the current state, it should terminate with an error. Else, if all actions could be executed, it should terminate with a "success" output.

2. Resource Collection 1 (40 points)

Set the initial state to be: the peasant is emptyhanded, the gold and wood tallies are zero and the capacities of all mines and forests are as above. Write STRIPS-like descriptions of the actions. (a) Set the goal state to be a gold tally of 200 and a wood tally of 200. Produce a plan and execute it in SimpleRTS. (b) Set the goal state to be a gold tally of 1000 and a wood tally of 1000. Produce a plan and execute it in SimpleRTS. In each case, output the total steps taken to actually execute the plan.

3. Resource Collection 2 (60 points)

Now suppose you have an additional action, BuildPeasant (the Townhall can execute this action). This action requires 400 gold and 1 food. The townhall supplies 3 food and each peasant currently on the map consumes 1 food. If successful, this operator will deduct 400 gold from the current gold tally and result in one additional peasant on the map, which can be subsequently used to collect gold and wood. Since your planner is a simple state space planner, it only produces sequential plans, which will not benefit from the parallelism possible with multiple peasants. One way to solve this is as follows. Write additional Move, Harvest and Deposit operators, Move_k, Harvest_k and Deposit_k, that need k peasants to execute and have the effect of k=1 to 3 parallel Moves, Harvests and Deposits, but will only add the cost of a single action to the plan. To execute such operations, your PEA should then find k "idle" peasants and allocate them to carrying out the Move_k, Harvest_k and Deposit_k operator by finding the nearest goldmine/forest/townhall to go to. Note that your PEA can further heuristically parallelize your found plans, though this reduction in cost cannot be accounted for by the planner. For example, with 3 peasants, suppose you have a Move_1(townhall,goldmine) and a Move_2(townhall,forest) in sequence. Your PEA can parallelize these actions to execute at the same time by noticing that their preconditions can be simultaneously satisfied. This sort of behavior by the execution agent falls under *scheduling,* a part of automated planning that we did not discuss in class. Be careful when writing heuristics for the BuildPeasant operator. Note that it has an immediate *negative* effect, i.e. it moves the plan farther from the goal. Somehow your heuristic needs to trade this off against the longer-term positive effect that the parallelism will allow.

Write a STRIPS-like description of the BuildPeasant action. Use the same initial state as above. (a) Set the goal state to be a gold tally of 1000 and a wood tally of 1000. Produce a plan and execute it in SimpleRTS. (b) Set the goal state to be a gold tally of 3000 and a wood tally of 2000. Produce a plan and execute it in SimpleRTS. In each case, output the total steps taken to actually execute the plan.

5. What to turn in

Prepare a ZIP file with a text/pdf file containing your written action descriptions and agent code (do not include any class files). Include a README with your name(s) and ID(s) and any other comments you have, such as special compilation instructions if any. Include both your names in the README if you worked as a pair for this assignment. Name your file as "yourname_PA3.zip" and use Blackboard to submit it. **This zip file should only contain your source code, readmes and makefiles, not executables/object files/data files/anything else, and must be timestamped by the due date to avoid a late penalty.**