AI Programming Assignment 2 (Max Points: 100).

Due is 24 April 2019.

In this assignment you will use probabilistic reasoning to solve a scouting/resource collection scenario in SEPIA. Download the pa4.zip file from the website.

1. Problem Setup

The scenarios you will solve are built around the "prob_19x25_e1" and "prob_32x32_e1" maps in SEPIA and the "probConfig" configuration file. In these maps, you start with a townhall and three peasants in ~~the lower left corn~~er. In the upper right corner there is a goldmine. The goal in this scenario is to collect 2000 gold. However, this map is only partially observable to you and your units. Somewhere hidden on the map are several enemy Towers. Towers are immobile units that shoot arrows with probability 0.75 at any unit in their attack range. If while moving your peasants come too close to a tower they will be shot and eventually die. (You can build more peasants if you have collected some gold.) Your goal is to collect the 2000 gold while losing as few peasants as possible (essentially by discovering where the towers are hidden and then avoiding them.) The Towers are controlled by the provided OpponentAgent.

To do this, your agent should maintain a probability distribution over each cell that describes the chance that it has a Tower. As your peasants move around, they collect observations. Cells that become visible and do not contain towers immediately will have probabilities of zero of having a Tower, and likewise, visible cells containing Towers have probability one. If a peasant moves to a cell and does not get shot, there *may* not be a Tower nearby. On the other hand, if it does get shot, there is a Tower nearby (note that a Tower's attack range (4) is larger than a peasant's sight radius so a peasant can be shot without being able to identify the location of the shooting unit). Further, Towers are rare: most cells do not have Towers. However, cells with forests are quite likely to have a Tower nearby somewhere.

At each step, your agent will have a *posterior probability* distribution that tells the agent how likely each cell is to have a tower. For each location to which it is possible to move, your agent needs to balance the risk of being shot by a Tower if it moves there (note that this is *not* the probability that the cell contains a Tower) to the benefit of being closer to the goldmine to collect gold, and pick a cell to move to that offers the best tradeoff. You can write this objective function as you see fit. Note that you may also want a term for rewarding a visit to a cell that may not be "closer" to the goldmine but will allow you to pinpoint a Tower's location. By repeatedly choosing the best location according to this objective, and updating the probability density as new observations are collected, your agent should be able to find a safe path to the goldmine. There are completely safe paths in each map.

Two additional maps are provided in the zipfile (named *demo.xml) that show you the actual locations of the Towers in the provided maps, so that you can verify that your posterior probability estimates are accurate and your agent is trying to do the right thing. Do not, however, tailor your code to these maps/Tower locations (e.g., by simply always choosing the safe paths from the start) as we will run your code with other maps.

2. 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. Name your file as “yourname_PA4.zip” and use Canvas 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.**