Alexandria University
Faculty of Engineering
Computer and Communications Program

CC482: Artificial Intelligence
Assigned: Thursday, August 1st, 2019
Due: Thursday, August 8th, 2019

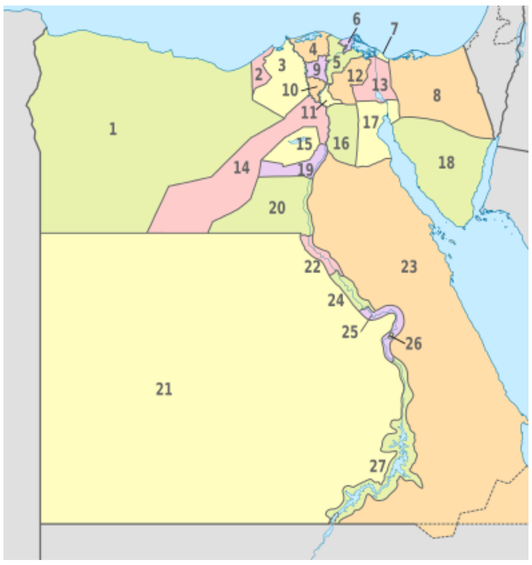
Assignment 2 - Risk Game

1 Overview

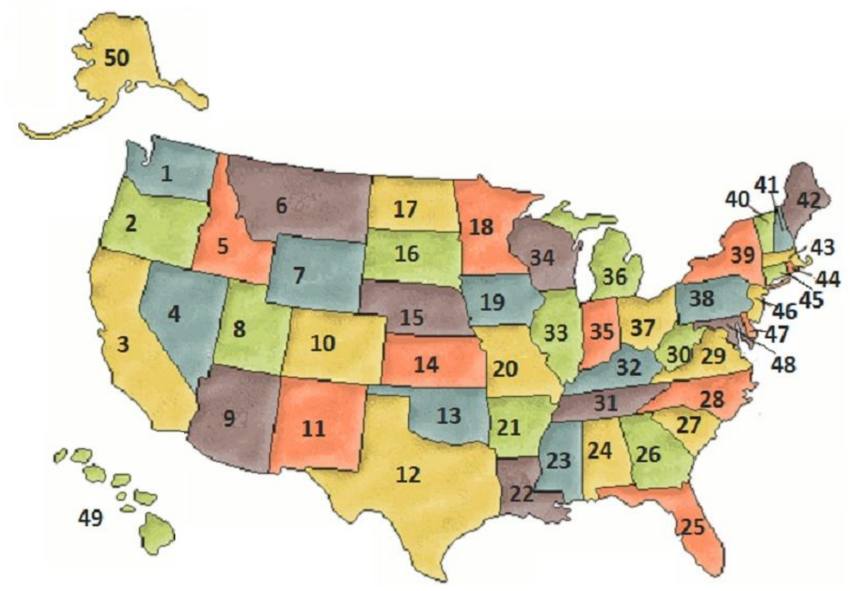
Risk is a popular game for major strategy type players. In Risk, the objective is to conquer the world by attacking to acquire territory and defending your own territory from your opponents. Risk is a turn-based game and is best if played with two to six players.
In this assignment, you will be developing an abstract and simple GUI version of the Risk game, but you need to understand the rules of the full version.

2 Abstract Risk Version

The Risk game board is divided into a set of territories based on the used map. Here, your game should provide two map options: Egypt, and United States.



(a) Egypt



(b) United states

At the start each player has 20 armies to be placed over the available territories in the board. Each turn consists of two parts: placing armies and attacking. At the beginning of your turn, count the total number of territories you own and divide by three. This is the number of additional armies you will receive for territory (ignoring fractions). If the number of territories divided by three is less than three, you receive three armies instead. After placing your armies you may declare attacks. You may only attack territories adjacent to your own, where your territory contains two or more armies. You may attack with any number of armies except for one, which must remain to defend your territory. There will be no fortifying step, for simplicity.

3 Playing Agents

You are required to provide four playing (Non-AI) agents, and four playing AI agents.

- Non-AI Agents:
  1. A human agent, that can make actions using the GUI.
  2. A completely passive agent, that places all of its bonus armies to the territory with the fewest armies, and doesn't make any attacks.
  3. An aggressive agent, that always places all its bonus armies on the territory with the most armies, and greedily attempts to attack territories with most armies that he can attack.
  4. A nearly pacifist agent, that places its armies like the completely passive agent, then conquers only the one territory with fewest armies (if it can).

- AI Agents:
  1. A greedy agent, that picks the move with best immediate heuristic value.
  2. An agent using A* search, with the same heuristic.
  3. An agent using real-time A*.
  4. A Minimax with alpha-beta pruning agent.

  You are free to choose a heuristic function that you find it suitable for the problem. Ties are broken by selecting the smallest numbered vertex among all that are equally good.

4 Game Modes

You are required to provide two game modes: Simulation mode, and playing mode. In the simulation mode, the user will choose any two agents from the defined agents except for the human agent, to play against each other. In the playing mode, the human agent will choose only one agent to play against it.
The initial placement of armies is done randomly for all players.

5 AI Agents Evaluation

You are required to provide an evaluation of the first three AI agents. The performance measure will be composed of two parts: L, the number of turns it takes the agent to win the game, and T, the number of search expansion steps performed by the search algorithm. The performance of an agent will be $P = f * L + T$.

Clearly, a better agent will have P as small as possible. The parameter f is a weight constant. You should compare the performance of the three agents against the completely passive opponent for the following values of f: 1, 100, 10000. Note that the higher the f parameter, the more important it is to expend computational resources in order to get a faster win!

For the Minimax agent, you need to provide test cases where your agent is able to defeat other agents, including the human agent.

Note: Fix the initial setting of armies for all the runs done in this evaluation part.

6 Bonus

In the abstract version described above, the random element was removed for simplicity. As a bonus your are required to include a defense random element with the following procedure:

1. The "attacker" may roll 1, 2, or 3 dice.
2. The "defender" may roll 1 or 2 dice.
3. The highest rolled attack dice is compared to the highest rolled defense dice, and
   (a) The attacker loses if the attack dice ¡= defense dice.
   (b) The defender loses if the attack dice ¿ defense dice.
4. The compared dice are discarded, and if each player still has dice left, repeat the previous step.

when an attacker losses, he losses all the armies he used in the attack to the defender.

7 Deliverables

1. Your well commented code.
2. A report showing your work and results. Also the report should contain the data structure used (if any) and algorithms, Assumptions and details you find them necessary to be clarified, Any extra work and Sample runs. You should show your algorithm and how it operate.

8 Notes

1. You may use Java , Python or C++ for your implementation.
2. You can work in groups of 4.

9 References

http://howdoyouplayit.com/risk-how-do-you-play-risk/
http://web.mit.edu/sp.268/www/risk.pdf

Good Luck