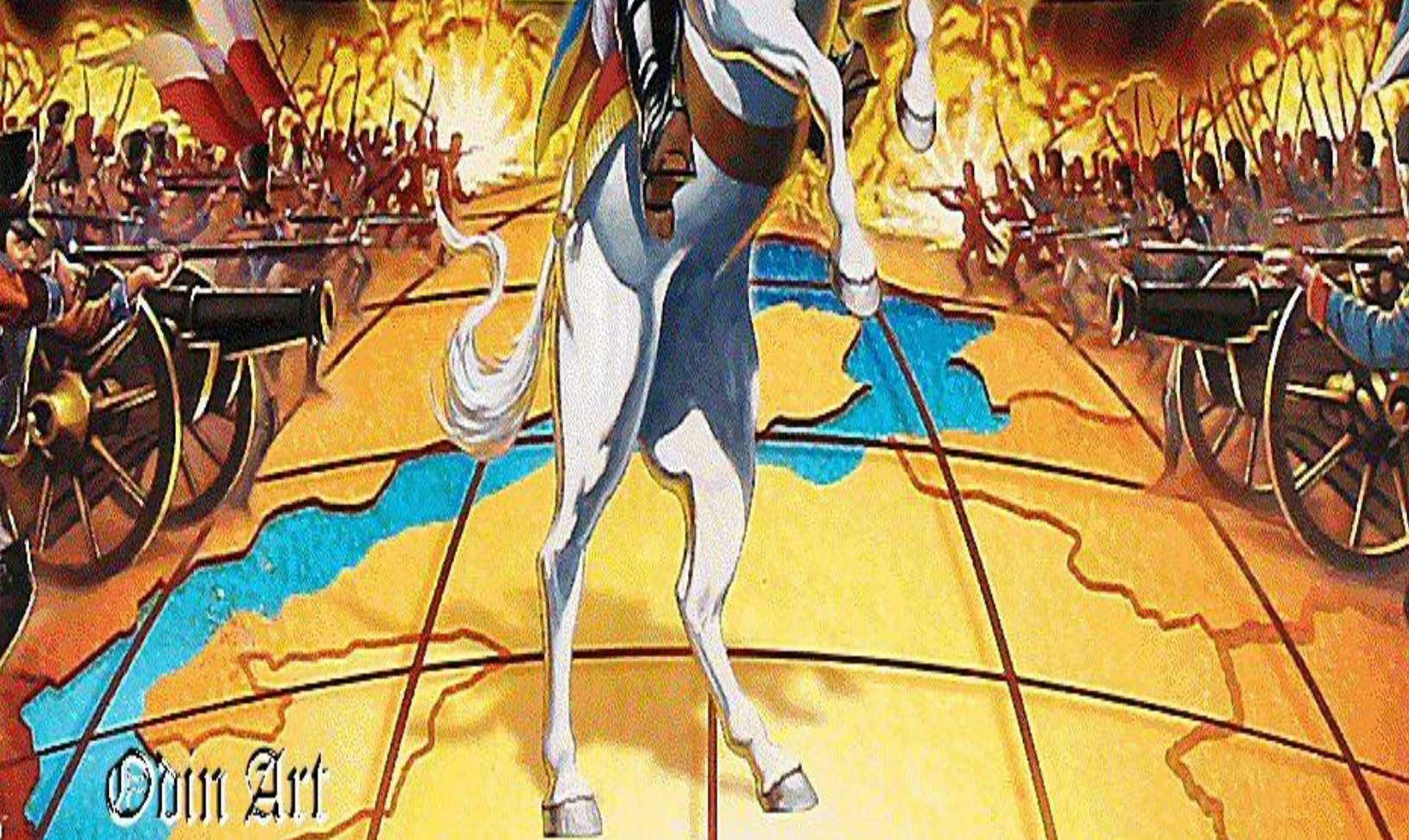


RISK



Omni Art

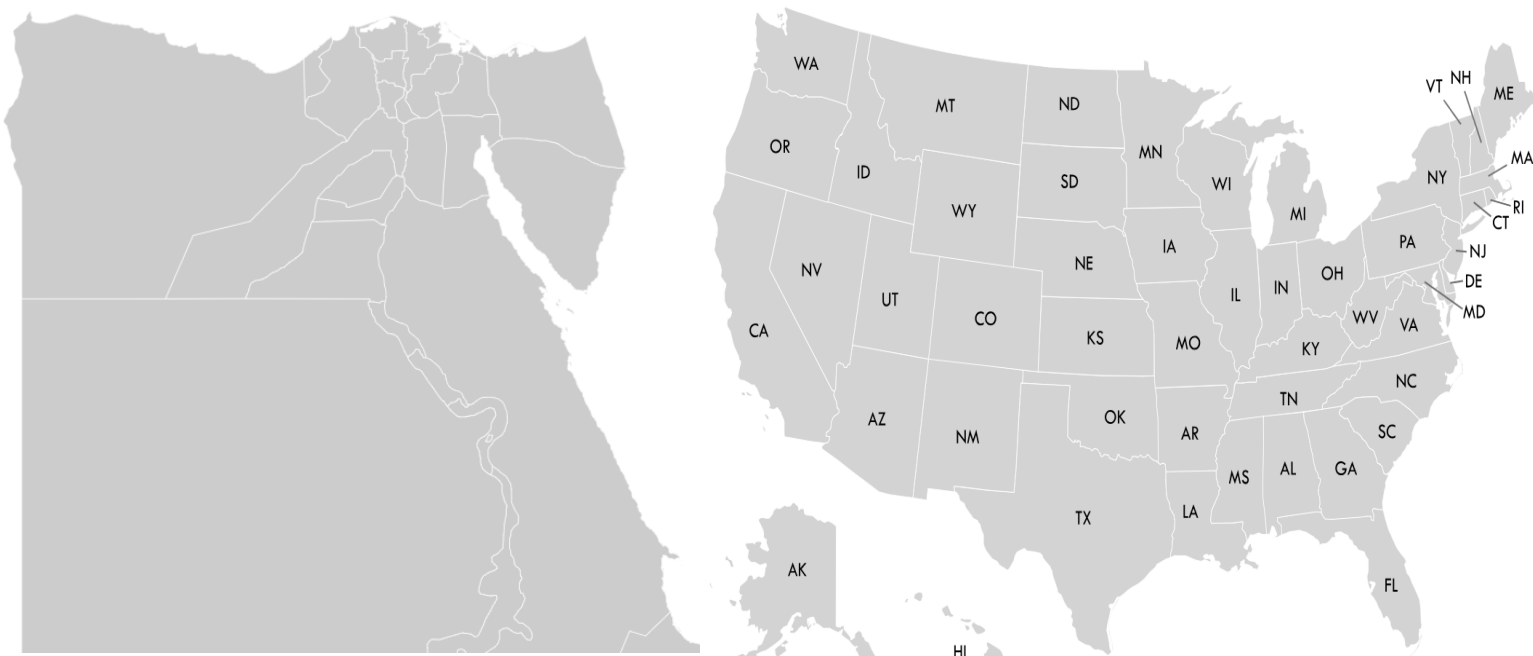
The Game of Global Domination™

Introduction

Risk is a complex board game produced by Hasbro that involves both luck and skill. The goal is simple: take over the world. Despite this simple goal, the game is very complicated and dynamic. Players attempt to take over the world by eliminating all other players. Players are eliminated when they lose all their troops on the game board. Players must be skilled in troop deployment and must be aware of the underlying probabilities present in the game.

The Board(s)

In Our game, we developed 2 options for choosing a map to play on it: Egypt & America...



Players may only move between adjacent territories, with the exception of the territories connected by red lines over water. The board can be simplified by turning it into a **GRAPH** where the territories are the nodes and the lines between nodes are the potential paths that can be taken from territory to territory.

One key to victory is control over all continents. Players that hold continents at the beginning of a turn get bonus reinforcements in an amount roughly proportional to the size of the continent (these bonuses will be detailed in the Rules section). Thus, the key positions on the board are the territories on the borders of continents. It is also important to know how to deal with the game board, as path efficiency is a key to success. It makes no sense to leave troops in the center of an area controlled by a player and choosing the quickest path from end to end of a continent is of the utmost importance.

Basic Data Structure

We used GRAPH as a basic data structure in order to implement each map (Egypt & America). Graph nodes represent territories and graph edges represent possible paths that can be taken through adjacent territories.

Organization of Code

The Code is divided into main regions:

1. Maps Region

Where we handle any basic data structure needed to whole game. It includes the following...

- Graph Class: The basic representation of any provided map. Graph nodes represent territories and graph edges represent possible paths that can be taken through adjacent territories.
- Node Class: Represent the basic component of any given node in our graph(map). These are the required components needed for any territory, i.e. Country name, Country ID, Player Owning that country...
- State Class: This is a basic useful data structure that is useful in A*, RealTime A*, MinMax Agents. Each state contains a snapshot of a given map grouped with some calculation in heuristics and costs.

2. Players Region

- Player Class: An abstract class holding the main requirements to any kind of implemented players, i.e. current map , enforcing soldiers, attack with rolling dice....

Non-AI Agents

- Human Agent Class: That can make actions using the interactive GUI.
- Passive Agent Class: That places all its bonus armies to the territory with the fewest armies and doesn't make any kind of attacks.
- 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.
- Pacifist Agent Class: That places its armies like the completely passive agent, then conquers only the one territory with fewest armies if it can

AI Agents

- Greedy Agent Class: That picks the move with best immediate heuristic value.
- A* Agent Class: That plays its turn based using A* search algorithm with the same heuristic and cost.
- Real-Time A* Agent Class: That plays its turn based using Real-Time A* search algorithm with the same heuristic and cost.
- MinMax Agent Class: That plays its turn based on alpha-beta pruning minmax algorithm.

3. Game Region

Where all requirement to a game begins. It includes all requirements for a game, such as: setting game map, setting kinds of players, initializing the board game in the beginning of each game...

This region is highly connected the GUI region.

4. GUI Region

Where all results from backends regions appears.

The GUI also handles the interaction of the human agent in case of being selected as one of the 2 players.

The GUI also handles the interaction of any computer agent (AI or Non-AI agents) to play together and display the result of each step on the screen.

Heuristic & Cost in AI Agents

- In case of reinforcement, we've chosen an **NBSR** (Normalized Border Security) heuristic. Its idea to find the owner's most weak soldiered that can be attacked and begin to add additional soldiers to make it more secure.
- In case of attack, we followed a strategy to choose the most soldiered country owned to this player and choosing the weakest soldiered to attack it and guarantee a high probability of success.
- As a Cost, we've chosen the depth taken to a node .



Extra Work (Bonus)

As extra work, we've implemented an Attack With dice version in our game....

We've completely implemented all cases in attack with dice like a real risk Game , including the following scenarios:

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 \leq 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 loss, he losses all the armies he used in the attack to the defender.

AI Agents Evaluation

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.

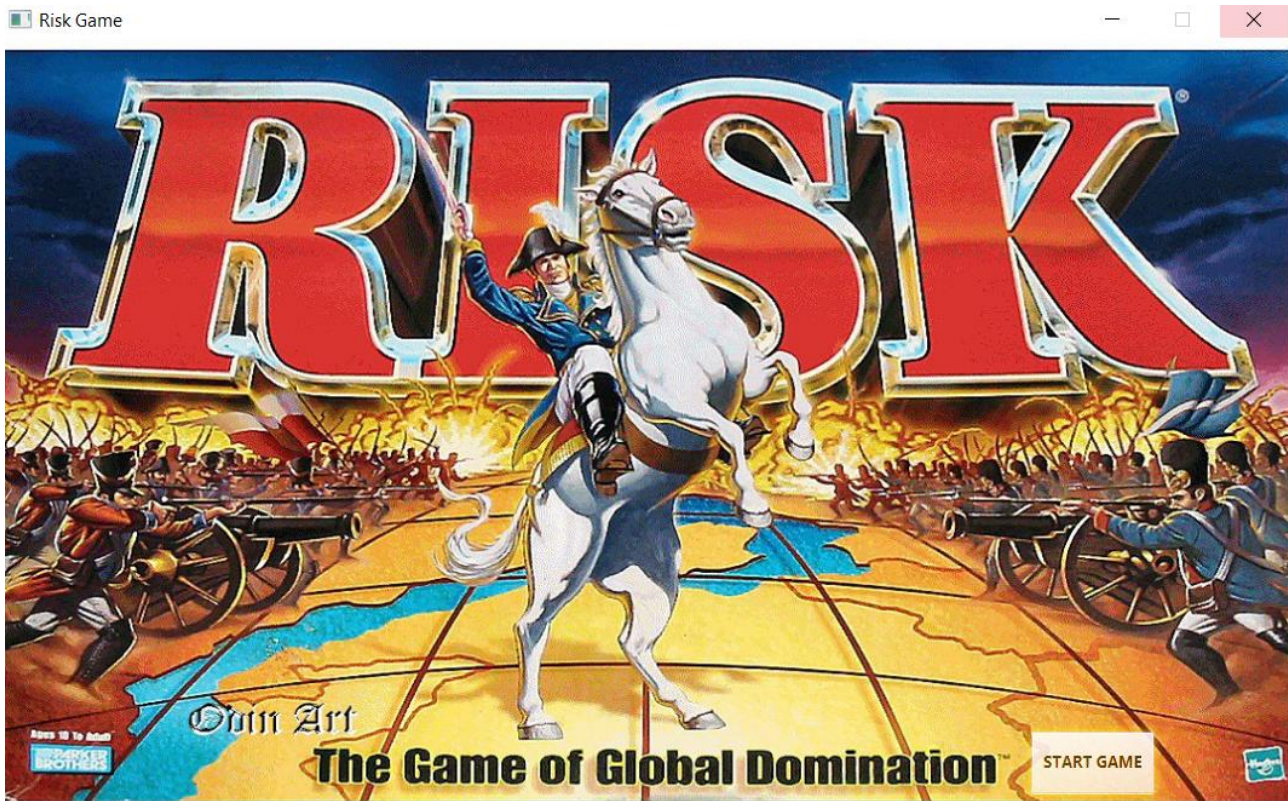
We compared the performance of the **first three AI Agents** against the completely **Passive Agent** for the following values of **f**: 1, 100, 10000.

The higher the f parameter, the more important it is to expend computational resources in order to get a faster win!

The following table was done using the simulation mode, where the user will choose any two agents from the defined agents except for the human agent, to play against each other....

f value	Greedy Agent Perf. L:35 T:10	A* Agent Perf. L:5 T:20	Real-Time A* Agent Perf. L:30 T:60
1	45	65	90
100	3510	520	3060
1000	35010	5020	30060

Sample Runs



Risk Game

CHOOSE THE PLAYERS AGENTS

CHOOSE PLAYER 1 AGENT

NON AI

☒ HUMAN AGENT

☐ PASSIVE AGENT

☐ AGGRESIVE AGENT

☐ PASIFIST AGENT

AI

☐ GREEDY AGENT

☐ A* AGENT

☐ REAL A* AGENT

☐ MINIMAX AGENT

CHOOSE PLAYER 2 AGENT

NON AI

☐ HUMAN AGENT

☒ PASSIVE AGENT

☐ AGGRESIVE AGENT

☐ PASIFIST AGENT

AI

☐ GREEDY AGENT

☐ A* AGENT

☐ REAL A* AGENT


☐ MINIMAX AGENT

CHOOSE THE PLAYING MAP



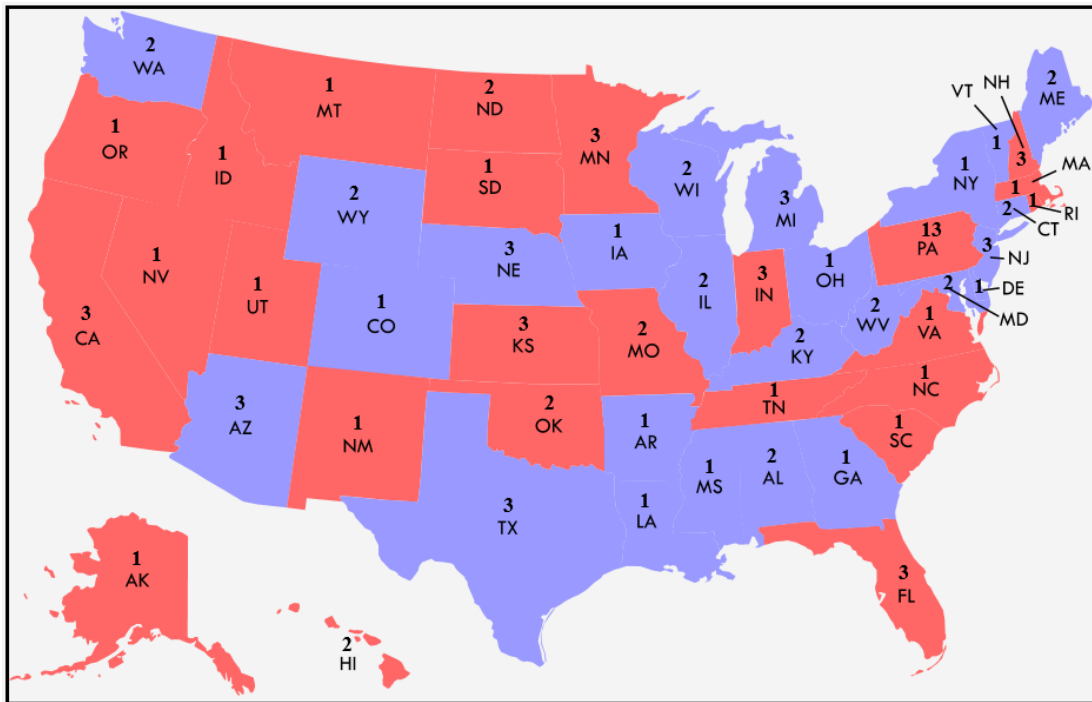
☒ EGYPT

☐ USA



START GAME

Risk Game

**PLAYER 1**

OWNED COUNTRIES

25**PLAYER 2**

OWNED COUNTRIES

25

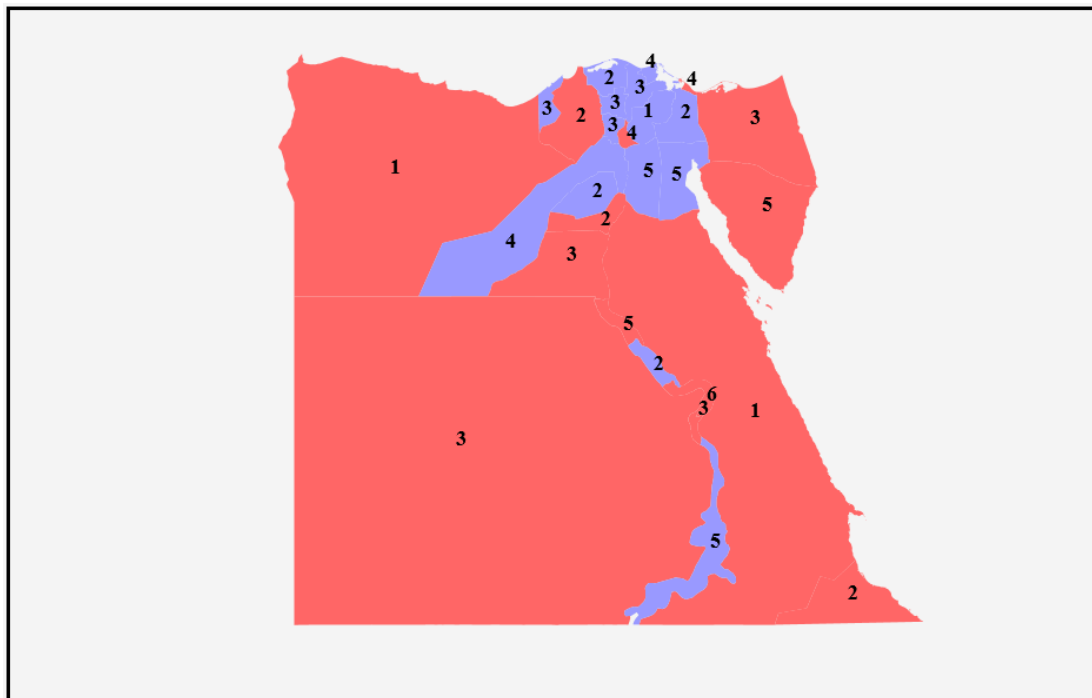
SECONDS LEFT:

PLAYER 1 TURN

SUBMIT

SOLIDERS ARE ADDED TO 'PA' NOW TIME TO ATTACK

Risk Game

**PLAYER 1**

OWNED COUNTRIES

15**PLAYER 2**

OWNED COUNTRIES

14

SECONDS LEFT:

PLAYER 1 TURN**ATTACKER WINS**

SUBMIT

YOU HAVE 5 ADDITIONAL SOLIDERS CHOOSE COUNTRY THEN TYPE NUMBER OF SOLIDERS TO ADD**Thank You**