Blokus with Artificial Intelligence

Thomas Mallinson, Jacob Peterson, Prasanna Gnanasekaran
Kansas State University
Manhattan, KS 66502

Abstract

Blokus (pronounced as "Block us") is an abstract strategy board game with colored pieces that players try to play on the board. Players are limited to playing pieces that touch at least one corner of their own pieces on the board. The goal of the game is to place as many pieces on the board as possible while blocking off the opponent's ability to paly their own pieces. Each of the pieces have different shapes and sizes, and each block within a piece counts as one point. The player who scores the highest amount of points wins the game.

Similar to other board games, Blokus contains strategies and patterns that can be solved with algorithms. In this project, we develop both random and greedy artificial intelligence agents to play one another, and analyze the effectiveness of different agents.

Introduction/Overview

Blokus is a strategy board game with transparent Tetris like shaped color pieces that players play on a game board. The main goal of the game is to place as many pieces on the board as the player can while blocking opponent's from placing pieces on the board as well. The game is very similar to games like Chess, Checkers, and Othello. The objective of this project is to create an agent that is capable of making intelligent decisions inside the scope of the game of Blokus.

The game can be evaluated at a basic level on its functional implementation of the heuristic-based game playing agents in order to always get the most points. Once the AI is functioning at a potent enough level for human-AI play, the evaluation criteria becomes much more straightforward. The AI should be able to defeat a human the majority of games, as well as employ strategies that are effective against a reasoning human opponent. The agent can also be evaluated by playing against itself, in order to quickly generate a large amount of analyzable data on the agent's performance. One final important criteria of note are the ability of the agent to make decisions, to score the most points. In our project, we will evaluate the agents playing each other [5].

Background

Blokus is a strategy board game invented by Bernard Tavitian and made its appearance in the market in the year 2000. The game was first released by a French company named Sekkoia. This strategy board game is one the most award-winning games in the 21st century. Since this game only has three rules, it is very adaptable for everyone across different age groups which increases its popularity level. Over 3 million copies of the game have been sold around the world. In addition, it has been compared to Go which is a great classic board game with a 2,000-year long history.

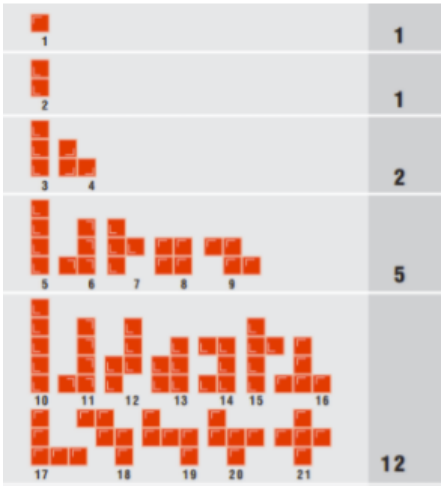

Figure 1.1: The 21 pieces that each player has at the beginning of the game.

How to Play

To understand how the game is played, it is first important to understand the layout and the pieces needed to play game. The following is the design of the game:

- A game board with 20 columns and 20 rows, 400 squares
- A total of 84 transparent game pieces where there is 21 pieces for each of the four colors. The pieces are sub-divided into the following:
- 4 pieces of monominos(1 square), 1 for each of the four colors
- 4 pieces of dominos (2 squares), 1 for each of the four colors

- 8 pieces of triominos (3 squares), 2 for each of the four colors
- 20 pieces of tetraminos (4 squares), 5 for each of the four colors
- 48 pieces of pentaminos (5 squares), 12 for each of the four colors

Blokus is known for the simplicity of the game, although it is easy to play it does require a good amount of concentration. The whole objective of this game is for each player to place as many of their 21 pieces on the game board. An average game should take about 20-30 minutes to play. The general overview of how a game works is as follows:

- The order of play begins with blue, then cycles through yellow, red, and green.
- The first piece played by a player must cover a corner square on the game board.
- Each new piece placed must touch at least one other piece of the same color, but only at the corners and not along the edge.
- There are no rules on how the pieces of different colors may contact each other. Once the piece is placed on the board it cannot be moved for the other turns.
- The game comes to an end when a player has placed all their 21 pieces on the game board or if all the players are blocked from putting down any more of their pieces.

These rules apply for any number of players, however, for the two-player version of the game, the board is a 14 rows by 14 column sized game.

Version of Blokus for Heuristic Study

For this study, the four-player version of Blokus with a 20 by 20 board size. This version allows us to compare four different agents that we designed with heuristics to reward strategies that real players use in the game.

The algorithm study focuses on the entire game play in general. The agents will primarily focus on maximum piece placements on the board. The two main algorithms in this study are random and greedy.

Random Strategy

The agent that utilizes the random strategy randomly selects a piece from a list of all possible moves and chooses a random placement to utilize it. The random strategy ensures unpredictability in the player's move, however, it also means the player may act sub-optimally and place lower priority pieces first. The algorithm is simple, but not a reliably effective agent to play Blokus.

Greedy Strategy

The agent that utilizes the greedy strategy places pieces based on the overall weight of different factors. Some pieces will be assigned a higher priority than others based upon those weighted factors. The piece with the highest weight will be chosen to be placed by the player. If the piece violates one of the constraints of the game such as overlapping or outside of the board's bounds, the next highest-weighted piece will be selected instead.

The factors that are used for this algorithm to weight pieces in this study are the size of the piece, the total corner difference between the players after a piece placement, and the distance from the center of the board.

Size Agent: The main focus of this greedy strategy is to select the piece with the largest size out of the available pieces and to prioritize playing it first. Placing larger sized pieces first is the fastest way to get higher scores, and one of the first basic strategies to Blokus. The wider available spaces also make larger sized pieces easier to place earlier as opposed to later, helping negate the risk of running into no valid piece placement.

Center of the Board Agent: This agent is designed to control the center of the board by weighting both the size of the pieces and the ones that can be played closest to the center of the board. Building towards the center of the board is another basic strategy in order to setup for a successful game. This is sort of a simplified version of the Barasona Opening [6] that is known to be effective in Blokus. The Barasona Opening is a starting diagonal board placement and cut off strategy that's designed to split the board to reduce mobility against other players.

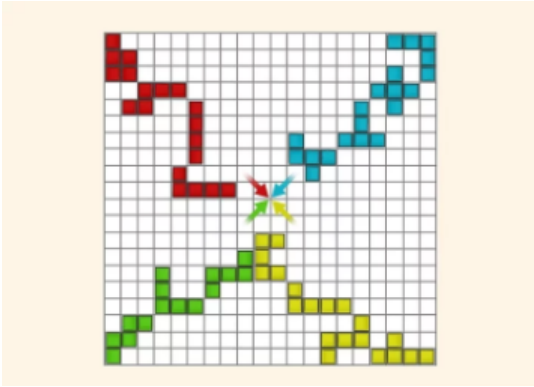

Figure 1.2: A general idea towards building towards the center of the board, an effective strategy in Blokus.

Corner Management Agent: The capability to place new pieces on the board for a player is partly determined by how many corners the players can try to find valid configurations for. The higher the number of corners available, the greater the chance of finding at least one valid placement. Cutting off other players' corners is a useful strategy, so these are taken into account with this greedy strategy. If a player has more valid corners placements than the opponents, they have an advantage in terms of possible board placement location and blocking opponent corners.

This strategy also takes into account the size of pieces in its weighting. Because the size of the pieces correlates to the possible final score more than corners, this strategy weighs size more than the total corner difference. An advanced greedy strategy maximizes the player's own score while keeping its total available corners higher than their opponents. This also increases the time complexity to run the algorithm.

Game Implementation

The project's Blokus simulation is developed in Python 3 using the Python Standard Library. The AIs can play Blokus using different strategies that are developed. [1]

Blokus Board

A Board class is defined to keep track of the state of the Blokus board. It updates the board for different players and can print the current board configuration to help debug or view game progress. The Blokus board is a 20 by 20 2-dimensional array that takes in characters to recognize piece placement. Each piece will be checked for adjacency, corners, overlapping, and whether it is in the bounds of the board.

Pieces

The Blokus pieces are developed under a Shape class with a different id to distinguish different pieces from each other. Each piece contains a list of points to determine to determine its attempted placement position on the board, as well as a list of corners for checking valid placement. A method is defined for each piece to set its points and corners using a reference point.

Players

A Player is defined to represent the four players in the Blokus game for this project. Each player has a unique ID to keep track. The players record their available pieces, corners they have to place those pieces on, and the heuristic strategy that they are implementing.

Each player creates a list of possible valid piece placements based on the current state of the board by going through every available piece, rotating and flipping them on each possible reference point, and then placing the piece based on the weights' of the specific algorithm. This was computationally costly, as there are generally a very high number of possible placements to iterate through.

Game

The Blokus class is defined to represent an instance of a Blokus game. It records the players in the game, the current round and game board state, and starting pieces given to the players. This class facilitates simulations of each round and performs the piece validation and placements for each player. It also determines when the game is finished by checking whether there are possibles moves to be made by any player in the game. Is no player can perform a move, then the scores of all players are compared and a winner is declared upon conclusion [3].

Results

To study the effectiveness of each of our agents, we simulated every combination of agents playing against each other, as well as each of the four agents playing against each other 15 times. We determined the win ratio between each of the strategies and recorded the data in a results document included with our report. The order of play was not taken into consideration in our simulations.

The following list is the matches set to play each other 15 times each.

- Random Agent vs. Greedy (Size)
- Greedy (Center Control) vs. Random
- Random vs. Greedy (Corner Management)
- Greedy (Center Control) vs. Greedy (Corner Management)
- Greedy (Size) vs. Greedy (Corner Management)
- Greedy (Center Control) vs. Greedy (Corner Management)
- Greedy (Center Control) vs. Greedy (Size) vs. Random vs. Greedy (Corner Management)

#	Agents	Winner	Differential
1	Random (P1 & P2) VS Greedy (P3 & P4)	Greedy	11
2	Center (P1 & P2) VS Random (P3 & P4)	Center	15
3	Random (P1 & P2) VS Corner (P3 & P4)	Corner	15
4	Center (P1 & P2) VS Greedy (P3 & P4)	Center	5
5	Greedy (P1 & P2) VS Corner (P3 & P4)	Corner	5
6	Center (P1 & P2) VS Corner (P3 & P4)	Tie	0
7	Center (P1) Greedy (P2) Random (P3) Corner (P4)	Center	2

Figure 1.3: The results of our simulations. The agents column shows which agents were playing against each other, while the winner column displays who won. The differential column is how many games the winner won by over the next closest finisher.

Reward Function

Since it proved to be difficult to implement alpha-beta pruning in a four-player Blokus game, our algorithm study focused on heuristics prioritizing the entire game play and maximum piece (points) placement on the board.

Our reward function is defined as the total number of wins across all simulations, with total number of points scored being a secondary reward function. This was done to reward agents that won their matches against other agents more often.

Y = Points + 25 * (Wins)

```
FINAL SCORES:
RANDOM: 5,910 pts + 2 wins = 5,960
GREEDY: 7,406 pts + 24 wins = 8,006
CORNER CONTROL: 8,131 pts + 38 wins = 9,081
CENTER CONTROL: 7,959 pts + 40 wins = 8,959
```

Figure 1.4: The final scores from our defined reward function.Greedy agents outperformed the random agent by far.

Summary

This case study on Blokus and strategies/heuristics used in the game was very informative. Our goal was to develop algorithms that were able to defeat a greedy oriented strategy. A greedy strategy can be viewed as a common strategy used by beginning players in the game, with our heuristics

attempting to simulate more advanced strategies. We found that our heuristics performed very well, and seem to be viable options even against advanced opponents/strategies.

While the center of the board control agent broke even with the corner control agent in testing, it outperformed the corner control agent by a decent margin when all four agents were present in the game. This leads us to believe that the center of board control agent acts as a great disruptor to strategies by limiting their moves in the late game. This seemed to be our most successful agent, and the one we have the most confidence in being successful outside of our controlled environment. The greedy agent also performed better than expected when thrown into an environment with all four agents present. We expect this is because the greedy strategy has a "survivalist" mentality and does a good job of maximizing its points early on in an environment with multiple other competitive strategies. Meanwhile, the random strategy performed very poorly against all other agents, suggesting both that Blokus is a game requiring a more precise level of strategy than checkers, and that the heuristics we designed performed at a solid enough level so as to never lose to a random agent. The greedy agent acted as a nice benchmark, and was even able to take a few games, likely because it gets rid of its large pieces early and wins the early game. Every agent that was designed performed well and helped to explain Blokus strategy as well as the success of various game-playing agent heuristic success rates.

We encountered several challenges during the development of this program. First and foremost was the time use barrier. The simulations we ran were quite costly in respect to time, which is one of the reasons we decided not to include time as a component of our reward functions and metrics for evaluating our agents. Another challenge that taught us a lot about AI and heuristics was the act of taking real-world game strategies and turning them into codable heuristics. This project truly became a study of generating creative, innovative, and efficient heuristics from board game strategies. This was perhaps the most rewarding aspect of working on this project, as it taught us a lot about how to implement heuristics. Another lesson learned during this process was adaptability. We had several setbacks and struggles throughout the process, from having to steer away from alpha-beta pruning to debugging, to running lengthy simulations. We are very proud of the final product, and hope it can be a great study on board games, heuristics, and game-playing agents.

Future Work

There are several modifications we would have liked to make to this project with more time and experience. First and foremost, utilization of the minimax algorithm with alpha-beta pruning, which technically possible in 4 player environments, is incredibly technical and difficult to implement. While we did research this possibility and found that it was outside the scope of our time and experience constraints, it would nevertheless drastically increase the performance of our system with respect to time. In a Blokus game with 4 players, a dozen or more turns, and each AI player being required to search through a list of all possible moves, alpha beta pruning would be a massive benefit. The largest roadblock to our system is the amount of time it takes to run a game for testing purposes. While it is still much faster than a game of four human players, alpha beta pruning is certainly at the top of our list when it comes to future improvements.

Another improvement we would like to add to our system is the ability to include human players along with the heuristic-driven agents we have created. While it is fascinating to observe the agent's performance against each other, it would be even more valuable to gauge the viability of these agents against intelligent human opponents. The act of implementing human playability into our Google Colab notebook would not be difficult as much as it would just be time consuming. The infrastructure is there to allow for human playability, and this would be something that we would want to implement in the future should we expand this project.

Finally, visual and interface upgrades would benefit our system. The game currently prints out the 2d character array, and while it works visually, it could be improved. Colors could be added so that the different player's pieces do not have to be represented by numbers, and there are several other visual improvements that could be made to the system. These changes are perhaps outside the scope of possibilities for Google Colab, which is why we decided not to pursue them further.

This project can have several applications. For one, it is a great case study on heuristic-based game playing agents. The performance difference between random, greedy, and more complex heuristics is on full display here, and put in a great context of a complicated board game. This project can also lay a foundation for future game playing agents, as well as for future Blokus AI improvements. If the ability for humans to play was added to this project, it would also act as a fun single player game in which a human player could practice Blokus or play for fun. This project was a great way to learn about game playing agents, heuristic, Python, and using creativity and innovation to design intelligent agents and heuristics for a board game.

References

[1] Blokus with AI. (2019, May 23). RandomlyUnique. http://randomlyunique.com/random/2019/blokus-with-ai/

[2] Cheng Cai et al. "From C to Blokus Duo with LegUp High-Level Synthesis." FPT2013 Design Competition Paper. http://legup.eecg.utoronto.ca/blokusFPT2013.pdf

[3] Chin Hung, Chao. 2018. California Polytechnic State University. https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1305context=cpesp

[4] Eskandari, N., Jahanshahi, A., Taram, M. (2019). Blokus Duo Game on FPGA. http://cseweb.ucsd.edu/ mtaram/blockus.pdf

[5] Lou, Anna. (2012, April 2). "Computer vs. Human: Exploring AI in the Game Blokus". California State Science Fair. http://csef.usc.edu/History/2012/Projects/J1415.pdf

[6] *Barasona Opening [Online* https://boardgames.stackexchange.com/questions/1497/what-is-a-good-opening-in-blokus