Counterintuitive Strategy for Application in Game Theory

New Mexico

Supercomputing Challenge

Final Report

March 30th, 2015

Team 126

School of Dreams Academy

Team Members:

James Edington

Noah Gelinas

Jonathan Daniels

Teacher:

Eric Brown

Project Mentors:

Creighton Edington

Nick Bennett

Talysa Ogas

Table of Contents

Executive Summary

The purpose of this project was to test the effectiveness of a counterintuitive strategy with potential application in game theory. The strategy was tested using the dominoes game “Muggins”. We wrote our own dominoes game in Python and programmed three different AI’s to play the game. The Random AI plays whatever dominoes it can at random. The Basic AI plays dominoes to intentionally score points. The third plays using our strategy, which takes into consideration several different aspects of the game at hand to make an effective decision. In order to get accurate data on the effectiveness of the different strategies, we tested each one against itself and against the others. Originally, we intended to have several thousand games for each strategy, but due to an unresolved code error we are currently only capable of running one hundred games at a time. In addition, due to time restrictions and unseen difficulties, the Counterintuitive Strategy has yet to be tested, although we fully intend to test it as soon as it is fully implemented. The results of what we were able to run show an impressive difference between the Random AI and the Basic AI. On average, the Basic AI won 98% of the time. Because The Counterintuitive Strategy incorporates the Basic AI, this data reinforces our hypothesis that our strategy, once tested, will be quite effective.

Problem Statement

The purpose of this project, as stated in the Executive Summary, was to test the effectiveness of a counterintuitive strategy and its potential application in game theory. The challenges of the project included devising the strategy, figuring out how to test it, implementing the strategy into our preferred testing method, and understanding the data produced.

Problem Solving Method

In order to test our strategy, we needed a medium. We decided to use the dominoes game “Muggins”, as it allowed for a comprehensive, realistic, and somewhat simple implementation. The easiest way to get data from this domino game was to write our own version that could play itself, using three different strategies. The first strategy, hereinafter referred to as the Random strategy, makes any move it possible can at random. The second strategy, referred to as the Basic strategy, makes intelligent decisions based on what choices it has in its hand. The third strategy, our Counterintuitive Strategy, looks at all aspects of the game at hand, including its possible moves, the moves that have already been played, the opponents potential moves, and the potential location of pieces throughout the game board.

When running the program, a desired number of games will be played and a comma separated value file will be exported in order to analyze results such as the winner for each game and the number of points scored. In the future we intend to export more information, such as the number of moves made per round and other Muggins specific data that will help us to understand why our strategy is or is not effective.

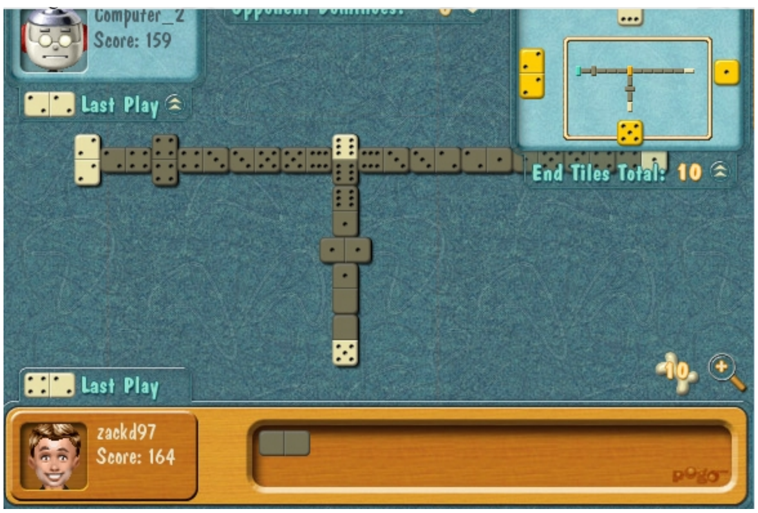
Application in Game Theory

Game theory is considered the study of strategic decision making. In detail, it is a study of conflict between intelligent decision makers. It has also been called decision theory. Game theory can be applied to countless fields, including biology, computer science, and economics.

This projects application in game theory lies in the potential use of the Counterintuitive Strategy in more settings than just Muggins. If the strategy proves to be effective, it may be applied to other areas such as artificial intelligence, other games, or even large scale conflict.

Muggins

The game that we used to test our strategy, Muggins, is a domino game based on scoring with multiples of five. Muggins, similar to card games, has players draw seven dominoes that they keep to themselves. The fourteen remaining dominoes are stored in the “boneyard” where they remain face down. The goal of each player is to empty their hand first and score 250 points. The first player (as decided by the owner of the highest valued double sided domino) places their desired domino on the board. The goal of the game is to have the open ended sides of the dominoes on the board equal a multiple of five, so if the first domino played is the (6,4), the player scores ten points. If the second player was to play a (5,4) connected to the (6,4), the total value of open sides would equal 11, therefore reaping no points for that player. Below is an image depicting what a standard Muggins game may look like.



In the image you will notice the end tile total value in the upper right corner. In this case, the last player to play scored 10 points. The highlighted dominoes are dominoes that are available to play off of. In order to play a domino, there must be a domino in play with a matching side. The player who's hand we can see in the image has no available moves, as the only domino in their hand is the (0,0), which must be played off an open end with a 0. That being the case, upon their next move, that player must draw dominoes from the boneyard until a playable domino is acquired. Once a player runs out of dominoes in their hand, they have “Dominoed” and will receive points equal to the total value of dominoes left over in the opponents hand, as well as play first in the next round. Rounds are played until a score of 250 is reached.

The Counterintuitive Strategy

When using our Counterintuitive Strategy, when applied to Muggins, the player focuses on forcing the opponent to draw while emptying their hand, in order to reap points equal to everything left in the opponents hand. By forcing the opponent to draw, the potential points received upon dominoing is greatly increased. If the player cannot force their opponent's hand, they will implement the Basic strategy of playing to score. The decision process for the algorithm is as so:

1) What dominoes are in my hand?
2) What dominoes are on the field?
3) What are the odds of other dominoes being in my opponent's hand versus the boneyard?

    1) How many dominoes are in my opponent's hand, and how many are in the boneyard?

    2) Has the opponent had to draw at any point in this game? If so, what were the end tiles?

4) Based on my observations, do I have a play that has the potential to block my opponent from playing?

    1) If so, how promising are the odds?

        1) If the odds are not promising, do I have an easy play that will score points?

    2) If not, what are my available moves? Can I score points?

By considering all of these questions, a player can get a very strong understanding of what plays they may have. Because there are twenty-eight individual dominoes in the game, it is possible to formulate an idea of where each domino may be. From the beginning, the player knows which dominoes are in their hand and which are on the field. If the opponent is forced to draw at any point during the game, the player can know for certain that any dominoes with end tiles that correspond to the open end tiles on the field are in the boneyard, as the opponent clearly has no possible moves.

This strategy is flexible. If there is no clear move to force the opponent's hand, it will adjust by playing to score. There is plenty of potential for the strategy to backfire, for example, if the player is forced to draw they may draw a domino that lets them score a high amount of points, although something like that could potentially be adjusted for after seeing whether or not it is a significant issue.

Results

Due to unforeseen complications in programming, we were unable to test the Counterintuitive Strategy at this point. We do, however, plan to test the strategy as soon as we solve our errors. Currently, the Counterintuitive Strategy AI is not fully implemented, and there is a bug in our code that stops us from running a very large number of games in one instance. That being said, we do have data showing the Basic and Random strategies against each other. As one may expect, preliminary data shows that the Basic vs. Basic results and Random vs. Random results show an almost equal number of wins and losses between two players over 100 games, where as the Random vs. Basic results show a very significant increase in wins and points scored, with the Basic strategy in favor. Once the bug has been worked out and the Counterintuitive Strategy AI has been fully implemented, we intend to test the Counterintuitive Strategy against itself and the other two, as well as run several thousand games of each variety in order to further solidify our data.

Conclusion

With what data we had, it was clear that the Basic strategy fully outperformed the Random strategy. One hundred games played showed a 98% win rate for the Basic strategy, with an average of a XX point difference between the Random strategy and the Basic strategy. We hypothesize that when the Counterintuitive Strategy is tested, it will outperform both other strategies. That being stated, we also hypothesize that the Counterintuitive Strategy will not score as consistently as the Basic strategy, but yield an overall higher average of points scored per round.

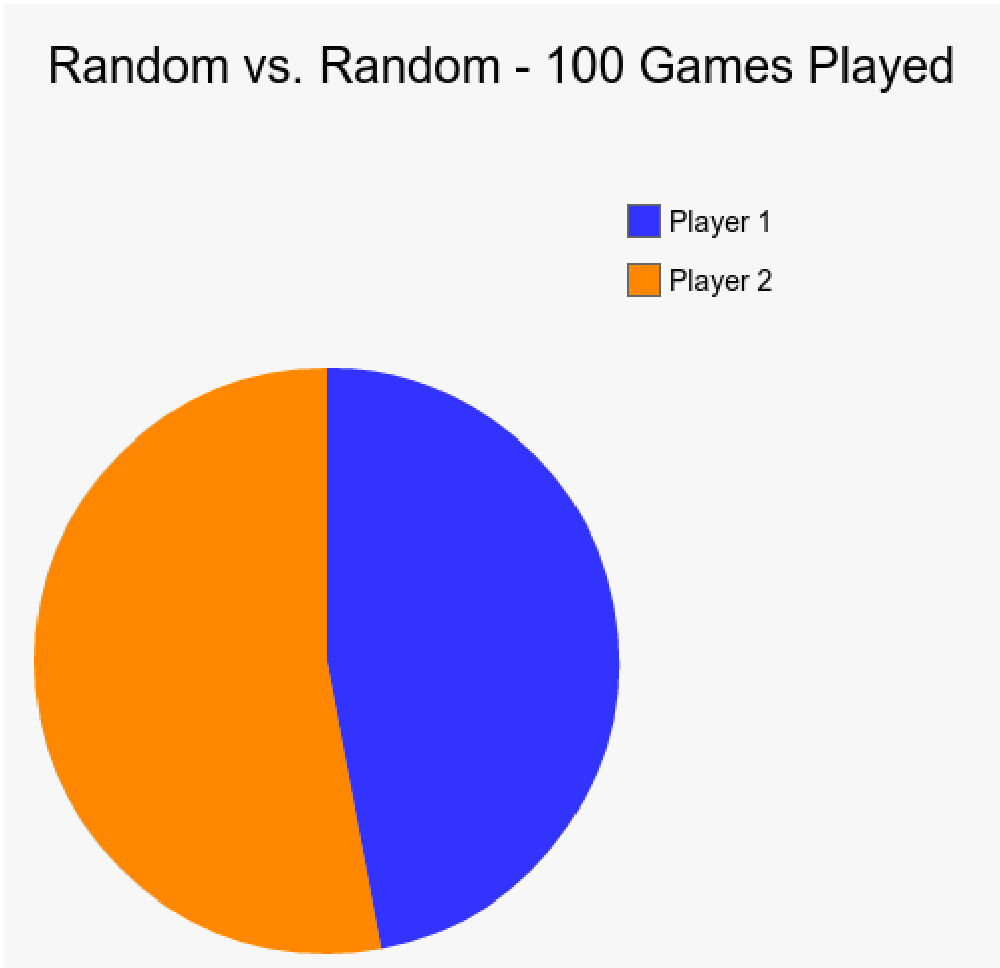
Personal Statement

As a team, we feel that the most significant achievement we accomplished was developing an understanding of game theory and informed decision making, and how it can be applied to computer science and artificial intelligence. We also gained some sweet domino playing skills, which was undoubtedly a bonus.

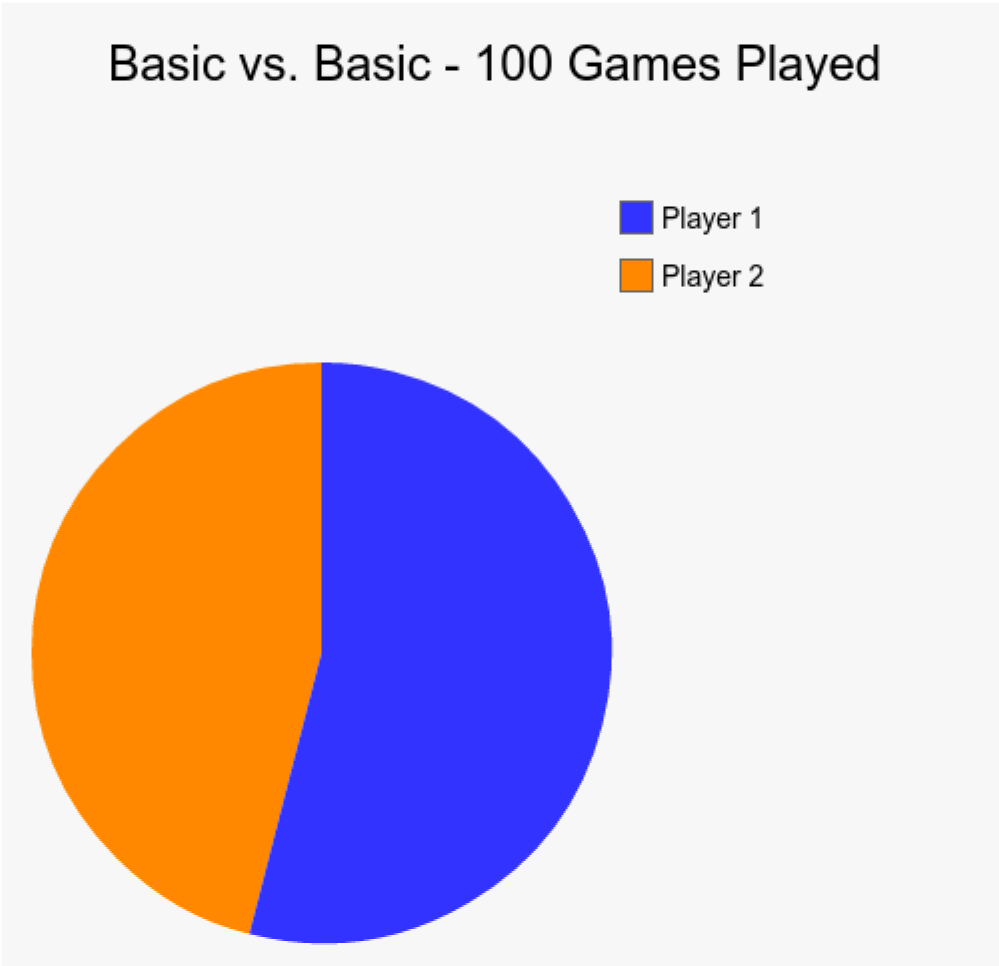
Acknowledgements

We would like to thank Creighton Edington, for helping us formulate this project and motivating us through it, Talysa Ogas, for helping us to familiarize ourselves with “Muggins”, and Eric Brown for acting as our sponsoring teacher. We would also like to thank Nick Bennett for giving us incredibly detailed and insightful advice. This project would not have been what it is without their help.

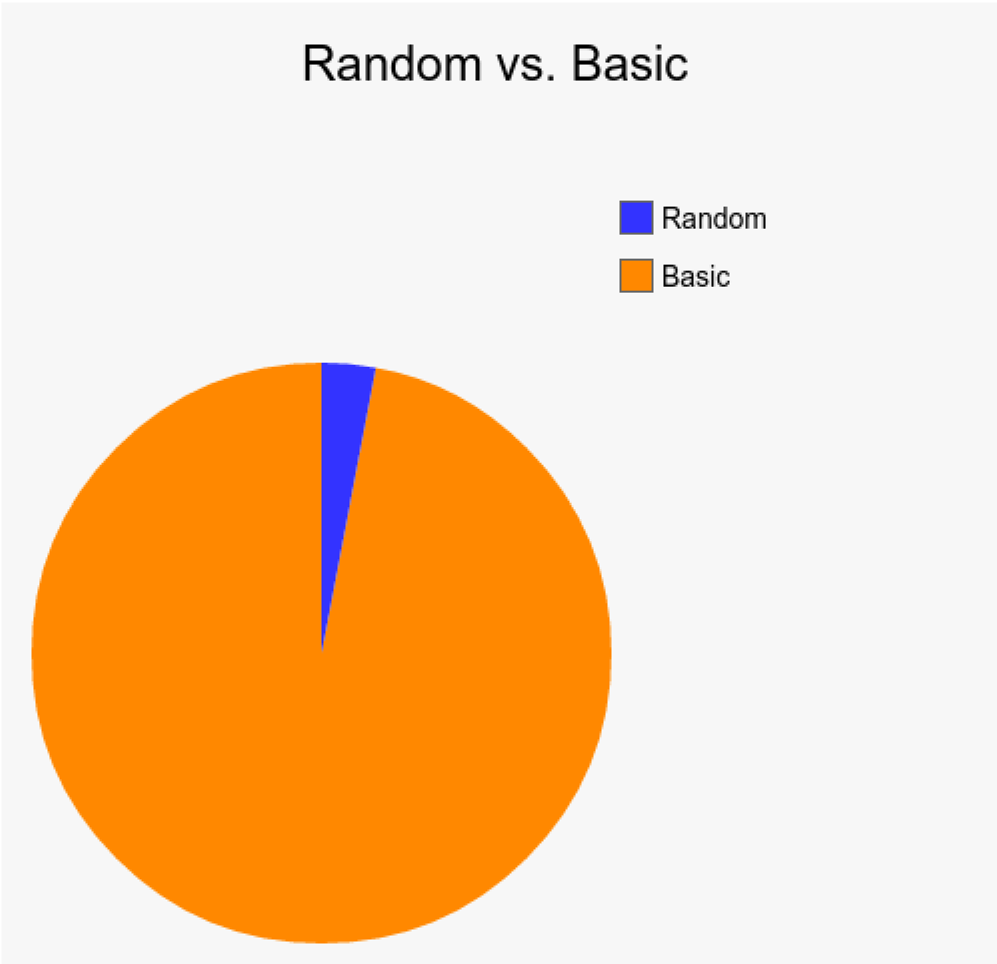
Graphs and Data



Player 1 - 47 wins vs. Player 2 - 53 wins



Player 1 - 54 wins vs. Player 2 - 46 wins



Random Strategy - 3 wins vs. Basic Strategy - 97 wins

Software

All code was written in Python 3, run on Linux machines, and hosted on Bitbucket. The website pogo.com was used for familiarizing ourselves with Muggins, as well as for the screen shot found above. All documentation was done with LibreOffice and Google Docs.

References

"Muggins Domino Rules - Domino-Games.com." *Muggins Domino Rules - Domino-Games.com*. N.p., n.d. Web. 30 Mar. 2015.

Wiens, Elmer G. "Egwald Operations Research - Game Theory - Introduction." *Egwald Operations Research - Game Theory - Introduction*. N.p., n.d. Web. 30 Mar. 2015.