Analysis of the Children's Game, "Guns," using Game Theory and a Genetic Algorithm, with the State Machine Approximation

Michigan State University

Aryan Gondkar

April 30, 2022

Contents

1 Abstract

1.1 About

The project analyzed the game commonly called "Guns", "007", etc using game theory and a genetic algorithm. It is a game involving three actions:

1. Defend: Protects user from potential attack for the round.
2. Attack: Attack opponent and use up one bullet. Wins the game if successful.
3. Reload: Increments bullet count for the user.

Summarized as a graphic:

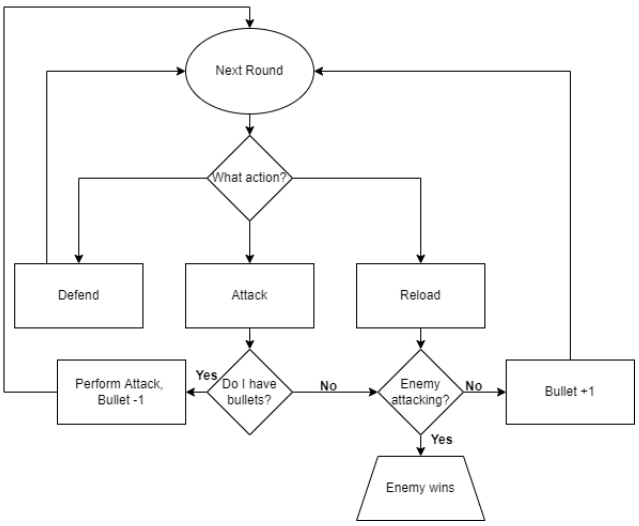

Figure 1: Flowchart summarizing game for a player

This flowchart is taken from the perspective of one player. Both players follow this chart to play the game. We also assume that each player knows how many bullets the other has.

1.2 Results

It was found that due to the problem of infinity, the algorithmic analysis gave results which although similar to the theoretical results, did have considerable deviation. It was also found that unless the opponents are trying to bring each other down, attacking is not a favorable strategy in S(n,m).

2 Introduction

Game theory is widely used to model competitive scenarios including diplomacy, board games, market decisions and biological competition. Its methodology boils down to:

1. Find pay-off matrix A
2. Find Nash equilibrium(s)
3. Decide on a strategy, and perform it

Most introductory texts on the subject deal with static payoff matrices. However, in this game, the payoff matrix changes depending upon how many bullets each player has. The paper will solve the simple case of the static matrices, broken into distinct states. Computer simulations can be used to assess the viability of this new state-machine approach.

3 Background

The payoff matrix refers to the gain player 1 can experience for a specific scenario a_{ij}. The strategy refers to the probability with which the player takes an action corresponding to the row for player 1, and column for player 2. If there is a probability of 1 for a specific action in a strategy, it is referred to as a pure strategy. Otherwise, it is a mixed strategy. A strategy in which no player can benefit by deviating from is called the Nash equilibrium. Every finite, non-cooperative game with more than two players has at least one pure or mixed Nash equilibrium [1]. This fact is why the state-machine approach can be considered viable.

The nxn payoff matrix is denoted by A, and the strategy employed by player 1 is written as an nx1 matrix x. Similarly, the strategy by player 2 is written as an nx1 matrix y.
Therefore, the expected gain G_1 for player 1 is given by following equation [2]:

$$x^T A y = G_1 \tag{1}$$

From here onward, Player 1 will be referred to as player, and Player 2 will be referred to as the enemy.

4 State Machine Approximation

Static payoff is as previously mentioned, the standard method used in the text (see [2] [3]). This paper will try to expand on that. Our attempt will include modeling the game as a state machine, which seems to be a new approach. The states will be as follows:

1. Player (0) Enemy (0)- Both the player and enemy have 0 bullets. Also referred to as S(0,0)
2. Player (n) Enemy (0)- The player has a positive number of bullets n, the enemy has 0 bullets. Also referred to as S(n,0)

3. Player (0) Enemy (n)- The player has 0 bullets, the enemy has a positive number of bullets n. Also referred to as S(0,n)
4. Player (n) Enemy (m)- The player and enemy both have some positive number of bullets, n and m. Also referred ti as S(n,m)

Each of these are elaborated in the Theoretical Analysis section. The state machine is implemented as follows:

1. Determine bullets with enemy and player
2. Determine state by bullet count
3. Choose appropriate strategy for the state
4. Play the round
5. Repeat from (1) until a winner emerges

A key thing to keep in mind is that this approach ignores how the states would affect potential outcomes in the other. It assumes that each state is independent of the other, and therefore the actions performed in one state do not affect the outcomes in the other states.

For example, it assumes that all bullets have the exact same value, regardless of how many are available. This is important in cases like S(n,0), where the best options include reloading or attacking. If the player only has a few bullets, it is worth conserving them. However, if the player has 10000 bullets, the reload action is pointless and it would be better to attack and hope the enemy does not defend. This is essentially the law of diminishing returns (see [4]). We ignore this factor in our approximation.

To obtain the specific strategies in a state, we have to calculate them beforehand. For this, we will assume 4 payoff matrices corresponding to each state, and then find the strategies based on that. This shall be done by the use of theoretical analysis, and the use of a genetic algorithm.

5 Theoretical Analysis

5.1 S(0,0)

This state is the one the game starts at. The payoff matrix is obtained below, and then simplified using the fact that neither can (or depending on the rules, should) attack the other.

		Enemy		
		Attack	*Defend*	*Reload*
Player	*Attack*	$(-,-)$	$(-,0)$	$(-,1)$
	Defend	$(0,-)$	$(0,0)$	$(0,1)$
	Reload	$(1,-)$	$(1,0)$	$(1,1)$

Ignoring the cells with dashes, we see a 2x2 matrix. From there, it is trivial to see that always reloading is the best strategy. This is in fact also the Nash equilibrium of this state.

5.2 S(n,0)

This state is our first interesting case. Our player has the ability to attack, but it might get wasted due to the enemy deciding to defend rather than reload. The payoff matrix is as follows:

		Enemy		
		Attack	*Defend*	*Reload*
Player	*Attack*	$(\infty, -\infty)$	$(-1, 0)$	$(\infty, -\infty)$
	Defend	$(0, -)$	$(0, 0)$	$(0, 1)$
	Reload	$(1, -)$	$(1, 0)$	$(1, 1)$

Ignoring the cells with dashes, we see a 3x2 matrix. Given that attack has an infinite payoff for the player (i.e., he wins the game), he should always want to attack. Knowing this, the enemy should choose to always defend, until the player runs out of bullets. However, we can see intuitively that this may not always be the best choice in the case the player has only a few bullets. As mentioned in the previous section, we shall ignore this situation for simplicity.

5.3 S(0,n)

This is just the complementary situation for the enemy. We will not discuss this section, as the same points as above apply.

		Player		
		Attack	*Defend*	*Reload*
Enemy	*Attack*	$(\infty, -\infty)$	$(-1, 0)$	$(\infty, -\infty)$
	Defend	$(0, -)$	$(0, 0)$	$(0, 1)$
	Reload	$(1, -)$	$(1, 0)$	$(1, 1)$

5.4 S(n,m)

This state is where standard techniques from game theory are the most relevant.

		Enemy		
		Attack	*Defend*	*Reload*
Player	*Attack*	$(-1, -1)$	$(-1, 0)$	$(\infty, -\infty)$
	Defend	$(0, -1)$	$(0, 0)$	$(0, 1)$
	Reload	$(-\infty, \infty)$	$(1, 0)$	$(1, 1)$

A distinct Nash equilibrium is visible here, the strategy (Attack, Attack). This state returns an interesting result in the algorithmic analysis, which shall be elaborated on in the respective section.

Again, we must keep in mind that we are ignoring the law of diminishing returns, essentially assuming we have an infinite supply of bullets in this state.

5.5 Results

Every game follows the loop of states $S(0,0) \rightarrow S(n,m) \rightarrow S(0,0)$ if we only follow the optimal (ie safety) strategy. This means that every game will continue

to forever, each player following the path $(Reload[0bullets], Reload[0bullets]) \rightarrow (Attack[1bullet], Attack[1bullet])$. Obviously, unless there is some risk-taking, no one can win this game. These results will now be investigated via a genetic algorithm.

6 Genetic Algorithm Analysis

The simulation will try to verify the results obtained above. It is also expected that it will give us a riskier strategy that would allow us to win, rather than continue the game perpetually. The algorithm will work as follows:

1. Determine the payoff matrices.
2. Perform equation (1). Player will try to maximize the gain G_1, whereas the enemy will try to minimize it. Similarly, there will be another payoff value for the enemy. (The minimization of the opponent's value is an interesting factor. This is studied in the S(n,m) section)
3. Select top players and enemies, mutate them by perturbing values in the vector.
4. Repeat from 2 until desired.

The GitHub link to the code is in the appendix (the code also contains the payoff matrices shown previously. These will be referred to as the "standard payoff matrices"). This approach is used as it allows us to explore a variety of solutions, and allows us to check the viability of a strategy by injecting it into the stack.

One key point to note however, is the **problem of infinity**. Infinity as a value does not behave well in this implementation. Because of this, we are forced to use very large numbers (VLNs) instead; which while a viable strategy, can give predictions much different that expected. For example, in the state Player (n) Enemy (m), (Attack, Attack) won't be the dominant strategy. This is because even in the case of a single Attack being successful, the gain jumps by infinity. However, with VLNs, (Defend, Defend) is usually the better strategy, as it guarantees the lowest loss, and therefore the highest gain (the exact calculations for this were performed by using [5].). This can explain much of the deviation from predictions we see below.

6.1 S(0,0)

This state's simulation used the standard payoff matrix indexed at 0. Unique solutions were not forced, and opponent gain minimization was rewarded.

Figure 2: Strategies obtained for S(0,0)

The expected strategy of always reloading has clearly won.

6.2 S(n,0)

This state's simulation used the standard payoff matrix indexed at 1. Unique solutions were not forced, and opponent gain minimization was rewarded.

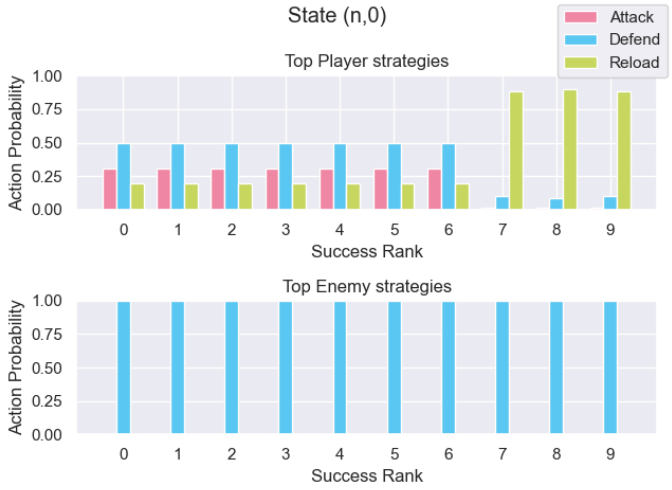

Figure 3: Strategies obtained for S(n,0)

The expected strategy of always defending has clearly won for the enemy, but

we see that our expected outcome of the player always attacking is not visible here. This is explained by the previously mentioned problem of infinity. The player experiences lower losses by spending less time on attacking, and using his move to reload (ie, gain a point) or defending (ie, not losing a point).

6.3 S(0,n)

This state's simulation used the standard payoff matrix indexed at 2. Unique solutions were not forced, and opponent gain minimization was rewarded.

Figure 4: Strategies obtained for S(0,n)

This case is the symmetric case for S(n,0) and will therefore not be elaborated on.

6.4 S(n,m)

This state is the one that yields the most interesting results upon varying the parameters. These are discussed below.

6.4.1 Simple run

This state simulation used the standard payoff matrix indexed at 3. Unique solutions were not forced, and opponent gain minimization was rewarded.

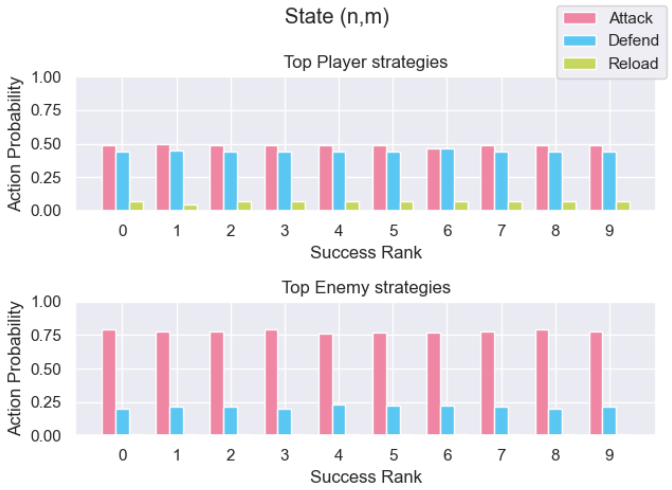

Figure 5: Strategies obtained for S(n,m)

We get close to the expected strategy of always attacking. There is a deviation caused by the problem of infinity; the player tries to make their opponent face a loss without cost to himself by sometimes defending. More interesting strategy pairs have been found by changing the seed, but those shall not be discussed here.

6.4.2 Run without minimizing opponent's gain

This situation is the most interesting, and perhaps even applicable to real life as a lesson. In this version, we run the simulation without trying to minimize the opponent's gain. Unique strategies are not being forced.

Figure 6: Strategies obtained for S(n,m) without antagonism

We see that the players simply focus on increasing their own score without trying to attack. In other words, reducing the incentive to harm the opponent allows both players to thrive. In fact, counter to the expectations, artificially and asymmetrically injecting an aggressive strategy does not grant an advantage. The aggressive strategy does not make it to the top!

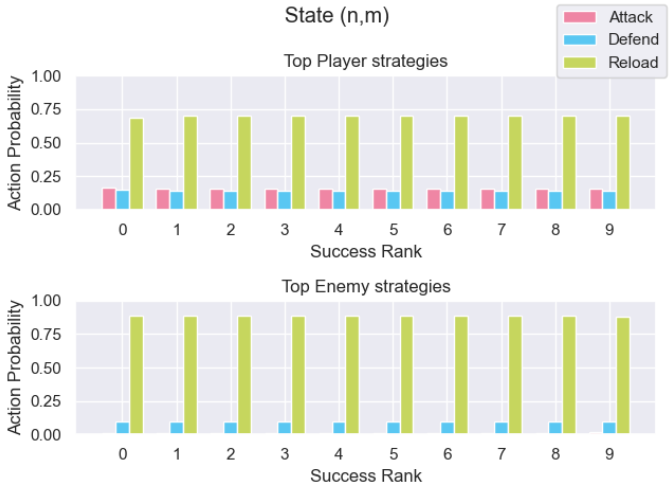

Figure 7: Strategies obtained for S(n,m) without antagonism, and an aggressive player injected

This can therefore be considered as proto-cooperation, as although there is no actual mutually beneficial exchange, there is much less active harm.

6.4.3 Results

As expected, due to the significant variation obtained in the strategies, it is impossible to predict a closed loop as was done in the case of the game theoretical analysis. However, this is actually a desirable feature, as in reality, no one values winning the game at infinity. Therefore, the strategies obtained via simulation are actually more realistic and should be more likely to succeed. These claims can be verified by further experimentation via simulated tournaments.

6.5 Conclusion

It was found that there was a strong correlation between the expected theoretical results and the results obtained by algorithmic analysis. The interesting case of a "friendly" duel was investigated, and it was found that attacking was not favored in that case. This investigation acts as a prompt for further investigation by varying the simulation parameters, game rules, etc. Particularly, the case of the "friendly" duel would be interesting to analyze as a tournament simulation.

7 Notes & Appendix

I would like to thank Professor Yiying Tong to allow me to investigate this topic for his CSE260 class's Honors option. The original proposal hoped to investigate dynamic pay-off matrices modeled with exponentials, along with bluffing and the evolution of cooperation in this game. However, the page limit forced the simplification of the investigation.

It was nevertheless a very enjoyable learning experience, and I hope to further explore this topic with competing neural networks with bluffing and the modeling of diminishing returns implemented, for senior project.

I would also like to thank Professor Kalyanmoy Deb for his guidance in my initial study of genetic algorithms.

7.1 Code

```
INSERT CODE HERE!
```

References

[1] "Nash's Theorem." From Wolfram MathWorld, https://mathworld.wolfram.com/NashsTheorem.html.

[2] Karlin, Anna R., and Yuval Peres. Game theory, alive. Vol. 101. American Mathematical Soc., 2017.

[3] Spaniel, William. Game theory 101: the complete textbook. CreateSpace, 2014.

[4] Investopedia. 2022. Law of Diminishing Marginal Returns. [online] Available at: <https://www.investopedia.com/terms/l/lawofdiminishingmarginalreturn.asp> [Accessed April 2022].

[5] Cgi.csc.liv.ac.uk. 2022. Solve a Bimatrix Game. [online] Available at: <https://www.cgi.csc.liv.ac.uk/ rahul/bi [*AccessedApril*2022].