Standoff: An Evolutionary Analysis

Dhawal Joharapurkar
Jessica Purser
William Sump

University of California, Santa Cruz

May 25, 2017

Abstract

This paper is an equilibrium analysis of Standoff, a stochastic game involving simultaneous move stages with state transitions. Players must decide their best response dependent upon their state, the states of the opposing players, and the states of players in the succeeding stage. In order to analyze Standoff, we constructed strategic form stage diagrams, bi-matrices for two- and three-player versions of Standoff, and determined a mixed strategy per stage of the game. To win, each player should make decisions in a manner that maximizes the probability of being the last man standing. A winning strategy is a complete action plan, which specifies a decision at each stage considering the state combination. Our analysis finds a symmetric equilibrium strategy at a given stage and uses evolutionary dynamics to ascertain the stability of our equilibrium.

Contents

1 Introduction

The game of Standoff is similar to rock-paper-scissors, but is more complex, requires additional strategic planning and has guns! Popular with young children, Standoff is a game made from the common movie trope where individuals are in gridlock with guns blazing and with fatal consequences. The depth of this game is characterized by means of stage transitions in which decisions are made simultaneously. The discrete-time game of Standoff meets all necessary conditions in order to be a stochastic game, a model developed by Lloyd Shapley in 1953 to describe a classification of repeated games with probabilistic transitions. Our game of interest is played sequentially through a series of stages with each beginning in some specified state. An alternative way of illustrating this is that at a given stage, the game encompasses a state that describes critical conditions of the players (the amount of bullets that they have, in the case of Standoff) that will transition to a new state at the next stage, contingent on the game's previous state and the actions chosen by its players.

The game requires two or more players, each with a strategy set of reload, armor, and shoot at a given state. A player can choose to reload in order to amass points and eventually acquire the upgrade gun, which solidifies their chances of winning. However, this strategy comes at the sacrifice of one's safety by not choosing to armor. This self-exposure may be potentially costly if another player reloaded in a prior stage, especially if the unarmored player has already reloaded as well. Thus, each player must consider the states of all other players when calculating their best response.

In order to analyze Standoff we constructed strategic form stage diagrams, bi-matrices for two-player and three-player versions of Standoff, and discovered a mixed strategy per stage of the game with a few exceptions. Standoff is a state-dependent game with simultaneous move stages with complex state transitions. It is more complicated than a repeated game, because it is dependent upon the outcome of the previous stage. In each new stage the best response is dependent upon the result of the previous stage; there is a clear dynamic relationship between consecutive stages. For example, the decision on whether to shoot would rely on how many bullets the player had accumulated from previous stages. If one player used their last bullet in the previous stage, then in the current stage, he or she is left with only the choice of armor or reload. To illustrate another example, certain outcomes will eventuate likelier targets amongst the players e.g. the player that has two bullets may try to upgrade while being shot at by another player.

The strategic tension of Standoff creates a payoff function whereby the relative attractiveness of using a strategy is greater when less players are using it. The greatest amount of strategic tension exists after the first stage (in which the null state players reload) when the players share the same individual state (i.e. 1 Bullet vs. 1 Bullet, 2 Bullets vs. 2 Bullets). After the first stage the strategies diversify not only due to the additional ability and strategy of the action shoot, but also because of threat of elimination and player actions conditional on previous, current, and probable future states. In other words, two-player Standoff is an elaborate duel. Players fire at one another in an effort to eliminate them while surviving themselves, and consequently must to choose the best response every stage in order to maximize their survival probability.

After the initial stage in the game each player has three available options: to reload, to armor, or to shoot one of the opposing players. Players are either eliminated or advance to another stage to make another decision. A new stage commences when there is more than one player remaining at the end of the previous stage. The conclusion of each stage results in each player being in one of five possible states: null, 1 bullet, 2 bullets, eliminated and winner. A winning strategy is a complete action plan specifying a pure strategy at each stage with regards to the state combination of all participating players.

The importance of studying such a game is the dynamic relationship between the strategic interaction of the players, the significance of individual and game states, and the use of hindsight and foresight (memory and strategic planning with probability) by successful players. It is a compelling examination of particular elements in strategic determination that we would expect to see in real life conflict. Then, from an evolutionary perspective, we see how a plethora of seemingly similar strategies with minute distinctions interact with each other, and how those slight differences in strategy can breed serious repercussions in a population with a massive strategy profile. These observations give us the strategy profile(s) that attract the largest shares and, if any, the profiles that undergo extinction. These discoveries will shed light on the evolutionary power of offensive, defensive, and advancive strategies in situations of high risk and immediacy. The purpose of this paper is to concretely explain these aspects of game theory by determining the unique symmetric Nash Equilibrium strategy profile for the two-player game of Standoff, where each strategy is a complete state-contingent profile, and then to expand our analysis of Standoff's complexity into the three-player game.

2 Model

The game commences at time (stage) $t = 0$ at state s_0, progressing through discrete stages at $t = 1, 2, ...,$ At stage t in state s_t, each player $i \in I$ is prescribed an action profile at from the finite set A, where $A = (R, A, S)$. Standoff's time-frame may continue indefinitely, such as if two players exclusively choose to reload as their action plan, which sends them through the same states, the same rewards, and gives us a simpler type of game satisfying the conditions of a Markov chain. However, given that such pure strategies are dominated and given a possible mixed strategy, a standard run-through of Standoff is likely to be finite and vary in its applications.

Additionally, we include the transition probability vector p from S x A, the state space and the complete contingency strategy set. Transition probability $p(s_t,\ a_t)$ determines the next state in the next stage s_t+1 relying upon the given current state and current action profile. The set of action profiles to state space S is denoted by $A = x_{i \in I_{Ai}}$. Payoffs are determined by the action plan at applied corresponding game G_{s_t} and are rewarded at the conclusion of the game.

Due to the complexity of the game, we will be employing a more specialized mathematical model of stochastic analysis, called an absorbing Markov decision process, which takes into account a Standoff player entering an immutable state (elimination). This framework provides a foundation to study decision making under circumstances where participants have only limited control of an otherwise random outcome.

2.1 Instructions

Standoff can be played with any amount of players, though we limit our analysis to a duel for simplicity and greater depth of exploration. The objective of the game is to avoid elimination and be the last player surviving. Players will need to decide on a response (Reload, and Armor, Shoot) before the beginning of each stage, and then simultaneously reveal their response on the count of three.

The null state in which a player has no bullets is the state that players begin the game, return to in the event of a draw, or return to when they have one bullet and choose to shoot. Players can only shoot if they have reloaded in a previous stage. The 1 bullet and 2 bullets states are resultant of a player reloading without being shot. The eliminated state of -1 is the result of player being shot during a stage. The winner state of 3 is the result of a player being the only survivor at the end of a stage, with losing players receiving a payoff of -1. Reloading three times permits the player an additional, and albeit most attractive, strategy. The bazooka, which is capable of eliminating all other competitors, ends the game with the players in the winner state if no other player upgraded in the same stage. During gameplay, a major concern is that choosing to reload leaves a player vulnerable to other players who choose to shoot, the probability of which rises the more bullets a target player accrues.

In the game Standoff the response to shoot is established by signaling a gun with one's hand and pointing it at a target, which has the outcome of elimination and is avoided if the targeted player chooses "armor." When a player chooses to shoot another player they leave themselves vulnerable because they cannot armor themselves. Otherwise unless the targeted player also shot at their shooter, the targeted player is eliminated. Two players shooting at each other results in a draw and the elimination of both players if other players are present in the current stage (I.e. the last player remaining in a three-player game is deemed the winner). If there are only two players in the game and they choose to shoot each other, they will both begin the next stage in the null state. Likewise if there are players shooting in a circular fashion (i.e., Player 1 shoots Player 2, Player 2 shoots Player 3, Player 3 shoots Player 1), then all players start over in the null state. The action of shooting consumes one bullet in addition to leaving the shooter exposed to other players and, subsequently, the likelihood of being shot by them during that stage. In the case that two or more players obtain a bazooka in the same stage, it will eliminate all other existing players and the upgraded players will enter the next stage in the null state.

The response to reload is signified by pointing one's gun hand in the air. Each stage that a player reloads allows them one bullet; a player can hold a maximum of two bullets at a time. Reloading past this capacity automatically upgrades the player's gun to a bazooka, which will allow the player to win the game regardless of how many players are left with the condition that no other player upgraded in the same manner within the same stage. If this occurs, only the upgrading players continue onto the next stage in the null state with all other players disqualified.

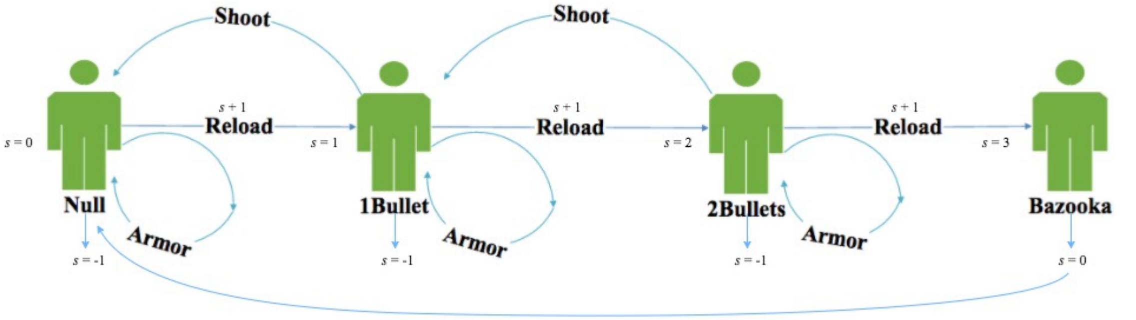
The response to armor is signaled by crossing one's arms over one's chest. Choosing armor bestows immunity upon the player for the concurrent stage. This response is essentially the safest option for any player to prevent their own elimination. However, armoring is not invulnerable to a bazooka, therefore playing a strategy of strictly armoring to avoid losing may eventually be ineffective and result in the player's elimination from the game.

2.2 Actions and States

The state space S of Standoff contains a finite set of individual state combinations. Individual states include null, 1 bullet, 2 bullets, eliminated and winner. For the two-player game, $S = (0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2), (3,-1), (-1,3)$.

The inclusion of immutable states in this state space accommodates the ambiguous resolution of the game. Although under natural conditions it is probable that Standoff will reach a conclusion with a winner and loser, the game also has the potential to be infinitely repetitive. With the presence of deterministic strategies, the state (0, 0) supports the possibility of a draw, or an infinite recursion of the actions in which both players return to the null state. When a draw occurs under these circumstances, the game is ultimately absorbed by and is never able to escape the null state. If either player wins, the game is in the absorbing states $s_t =$ (3, -1) or (-1, 3), defining the termination of the game.

To clarify, the game hypothetically never reaches (3, 3) or (-1, -1). The actions in the previous



stage define the state the game transitions to, so the actions that would result in a draw immediately move the game to (0, 0). However (0,0) may still be a transitory state when a draw has not occurred, such as when the game initially begins or as in the situation at (1, 0) when a player shoots at the armor of the null-state player. The game moving between these nine transitory states imply it is still ongoing. Thus there is a total of 11 possible states the game may move to, with three of those states being immutable and designating the consequence of two strategies in play.

The stage diagrams provided in sections 2.3 and 2.4 illustrate the inherent dynamism of Standoff. The arrows indicate the game's change or recurrence of its state as it transitions through discrete stages. The diagram including all strategies demonstrates the game's potential to shift suddenly between its many states in a given stage. At a glance in both cases, 1 Bullet vs. 2 Bullets offers the highest amount of directions the game may take.

Strategies that result in a draw (an infinite repetition of the game with no winner) originally corresponded with the payoff 0 for both players. In other cases when two strategies concede a finite game, the winner received a payoff of 1 and the loser -1. In order to proceed with Standoff's replicator dynamics, the payoffs assigned to these outcomes must be normalized to a 0 to 1 inclusive range, using Min-Max scaling:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

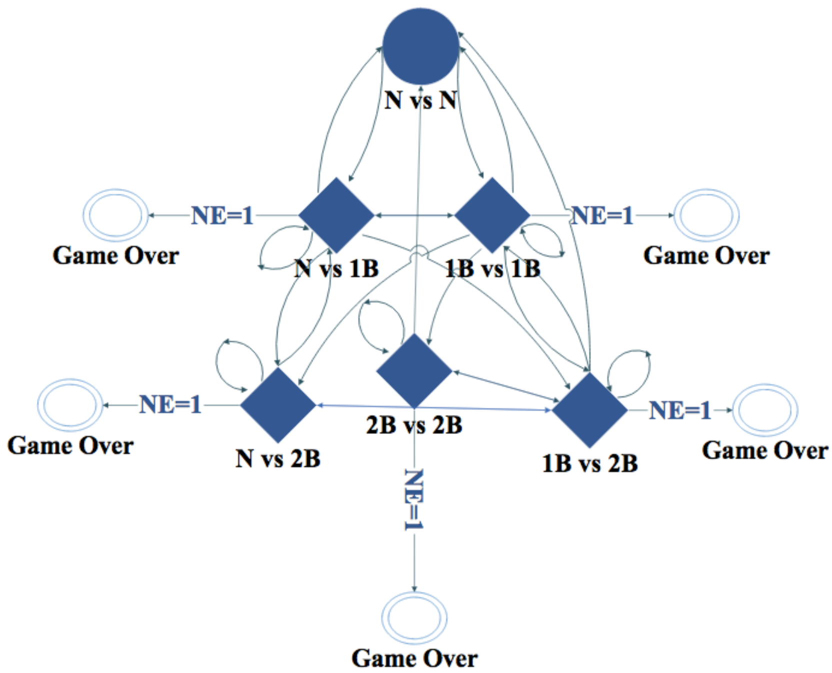
$$X_{norm} = \frac{X - (-1)}{1 - (-1)}$$

$$X_{norm} = \frac{X + 1}{2}$$

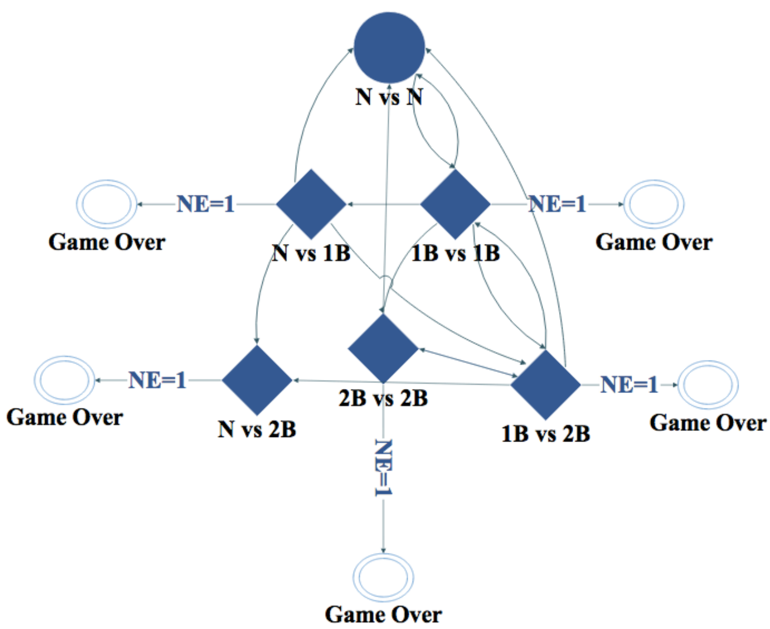
This provides the values of 0 for the loser, $\frac{1}{2}$ for a draw, and 1 for the winner in the computation of the game's replicator dynamics. In terms of the end states in S, 3 returns to a payoff of 1, -1, returns to 0, and 0 returns to $\frac{1}{2}$.

It is worth mentioning that there are some recursive scenarios in the game that do not return to (0, 0). With the use of deterministic strategies, a single game of two-player Standoff that returns to any of the transitory states for a second time elicits a repetition of that game's sequence to that point. With no winner or loser in an infinitely looping game, it is therefore a draw. For example, Player 1 may shoot an armored Player 2 at (1, 1), transition to (0, 1) with both of them reloading to (1, 2). Then, Player 1 armors and Player 2 shoots in this situation, bringing the game back to (1, 1). This is overcome in the code because of the use of finite sets of strategies. If the simulation has run through every state's action in a profile, it halts the loop and designates a draw for the situation where those two strategies face each other.

2.3 Stage Diagram Two Players, All Strategies



2.4 Stage Diagram Two Players, Non-Dominated Strategies



3 Simulation

Including dominated strategies, the two-player game has $2^3 \cdot 3^6 = 5{,}832$ possible strategy profiles, encompassing all possible actions a state state in any stage. This figure is representative of the two available actions (Reload and Armor) in the three states where the player in consideration is in Null, and additionally all three actions (Reload, Armor, and Shoot) in the remaining six state combinations:

State	Available Strategies
Null vs. Null	*R, A*
Null vs. 1 Bullet	*R, A*
Null vs. 2 Bullets	*R, A*
1 Bullet vs. Null	*R, A, S*
1 Bullet vs. 1 Bullet	*R, A, S*
1 Bullet vs. 2 Bullets	*R, A, S*
2 Bullets vs. Null	*R, A, S*
2 Bullets vs. 1 Bullet	*R, A, S*
2 Bullets vs. 2 Bullets	*R, A, S*

Figure 1: Note that a player cannot shoot when they do not have bullets, hence the exclusion of the action Shoot while in the null state.

Figure 2 provides the non-dominated strategies resulting from backwards induction and the iterated deletion of strictly and weakly dominated strategies. This produces a total of 324 possible pure strategy combinations. It is interesting to note that Reload is included as a viable strategy in each of the nine states except in Null vs. 2 Bullets, an observation that will be key in our analysis later.

The most effective method of analysis was initially discovered through the results of running the simulation with all 324 profiles, then consequently reducing the tested strategy space to the fittest twelve, then from that the fittest six profiles. The outcome of this preliminary testing made apparent the appropriate procedure for the analysis: a tournament-style elimination of strategies as per Axelrod's approach in *The Evolution of Cooperation*. Essentially this requires removing from

States	Undominated Strategies
Null vs. Null	*R*
Null vs. 1 Bullet	*R, A*
Null vs. 2 Bullets	*A*
1 Bullet vs. Null	*R, S*
1 Bullet vs. 1 Bullet	*R, A, S*
1 Bullet vs. 2 Bullets	*R, A, S*
2 Bullets vs. Null	*R*
2 Bullets vs. 1 Bullet	*R, A, S*
2 Bullets vs. 2 Bullets	*R, A, S*

Figure 2: Iterative deletion of strictly and weakly dominated strategies produces the listed rationalizable actions, dependent on the current state of the game.

the next run of the simulation the strategies that converged to zero (or approaching it, i.e. values to the negative three- hundredths power). Otherwise, share values of statistical significance made it to the next round. Therein replicator dynamics is repeated over an equal distribution for the remaining strategies. Intuitively, this is fitting considering the nature of Standoff in itself. With the incorporation of these 324 strategy profiles into the coded simulation, we find that following this tournament process leads to the convergence of four evolutionarily stable strategies.

3.1 R: Generating the Strategy Set

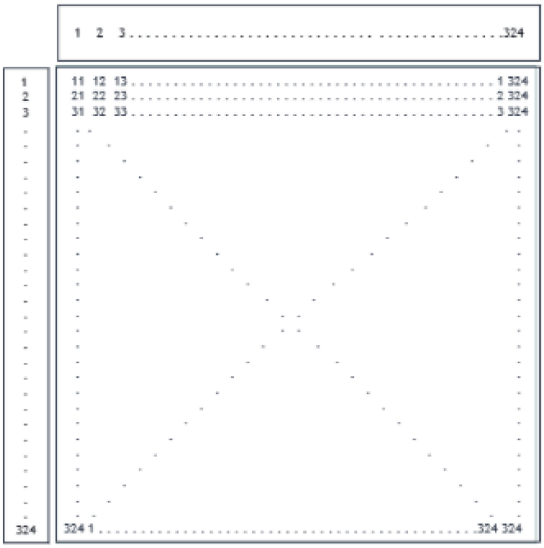
From the complete strategy set of 5,832 profiles, the programming language R was used to generate the 324 permutations of strategy profiles of interest, using as parameters the inclusion of only the non-dominated strategies per state as represented in Figure 2. This produces an array of the non-dominated strategy profile depicted in Figure 3, the values of which are imported into the Standoff simulation written in Python.

3.2 Python: Execution of Replicator Dynamics

Procedurally, the simulation of Standoff entails coding the two-player game in Python, a programming language popular for its wide range of applications. Therein the strategy set from R is imported into the Python program to run the simulation of the 324 vs. 324 profiles. In other words, we pitted the strategies against each other to output 104,976 payoff elements in the consequent 324x324 matrix.

The Python code designates the states of the game, wherein the simulated players' strategy profiles are limited to the non-dominated actions per state as per the results found using R.

With this large an array of strategies, the task of computing and observing the shares of all 324 strategies, their fitnesses (in regards to all 324 profiles), the average fitness, and then the consequent 324 dot products per period for 1,000 periods is accomplished by loops and appending commands written in Python code (refer to the Appendix). Each period takes into consideration the payoffs of the strategy matrix, as depicted below in the figure.



Algebraically, replicator dynamics for Standoff follows the process:

At stage t_0:

$$w_1 = s_1(t_0)w_{11} + s_2(t_0)w_{12} + ... + s_{324}(t_0)w_{1,324}$$

$$w_1 = \sum_i^{324} s_i(t_0)w_{1i}$$

$$w_2 = \sum_i^{324} s_i(t_0)w_{2i}$$

$$\vdots$$

$$w_{324} = \sum_i^{324} s_i(t_0)w_{324i}$$

To compute replicator dynamics at stage t_1:

$$s_1(t+1) = \frac{W_i}{\bar{W}} s_i(t), \quad i = 1, 2, ..., 324.$$

$$s_1(t_1) = \frac{s_1 w_1}{\sum_i^{324} s_i w_i}(t_0)$$

$$s_2(t_1) = \frac{s_2 w_2}{\sum_i^{324} s_i w_i}(t_0)$$

$$\vdots$$

$$s_{324}(t_1) = \frac{s_{324} w_{324}}{\sum_i^{324} s_i w_i}(t_0)$$

$$\vdots$$

$$s_{324}(t_{999}) = \frac{s_{324} w_{324}}{\sum_i^{324} s_i w_i}(t_{998})$$

Adhering to the tournament structure for replicator dynamics, convergence was achieved after six rounds of systematic elimination. The contest of the fittest profiles proceeded as follows:

Round 1: 324x324

Round 2: 76x76

Round 3: 26x26

Round 4: 19x19

Round 5: 18x18

Round 6: 4x4

4 Results

Using a uniform distribution, the replicator dynamic functions were rendered in Python over 1,000 and additionally 10,000 iterations beginning at t_0 to find the main basin of attraction. Observing the changes of these 324 shares under these conditions, Strategies 95 and 96 emerge as the main basins of attraction, holding equivalent shares of the population.

Of the 324 strategy profiles, 76 emerged with a significant share of the population. The four highlighted here are the strategies that appear in the final round, and consistently ranked highest amongst the majority of shares through the progression of the tournament. The strategies are ranked by their respective share values, largest to smallest, at the conclusion of evolutionary replication:

Rank	Strategy	Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
1	96	0.034272	R	R	A	R	R	A	R	R	S
2	95	0.034272	R	A	A	R	R	A	R	R	S
3	2	0.034011	R	R	A	S	S	S	R	S	S
4	93	0.028827	R	A	A	S	R	A	R	R	S
5	83	0.02877	R	A	A	R	R	S	R	R	S
6	108	0.028768	R	R	A	R	R	R	R	R	S
7	107	0.028768	R	A	A	R	R	R	R	R	S
8	3	0.027419	R	A	A	R	S	S	R	S	S
9	1	0.027419	R	A	A	S	S	S	R	S	S
10	94	0.026163	R	R	A	S	R	A	R	R	S
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
76	69	0.001277	R	A	A	S	R	R	R	A	S

Figure 3: For the complete ranking of the surviving 76 strategy profiles, see Appendix.

Two hundred forty-eight of the strategy profiles became obsolete by t_{999}, indicated by those shares' convergence to zero. These shares were subsequently eliminated from the simulation for the succeeding round matching the 76x76.

In the second round of replicator dynamics, the highlighted Strategies 93, 94, 95, and 96 are the predominant four profiles, with 95 and 96 again tied as the highest shareholders amongst the population. This round concluded with 26 strategies retaining share values. An additional 50 strategies were eliminated and therefore a total of 298 strategies were removed from the simulation. The 76x76 simulation in Round 2 narrowed our focus to the 26 strategies:

Thereafter, in Round 3 matching the 26x26 strategy profiles, the simulation was further successful in eliminating 7 more strategies (totaling 305 extinct strategies) and revealed the following:

Rank	Strategy	Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
1	96	0.08622801	R	R	A	R	R	A	R	R	S
2	95	0.08622801	R	A	A	R	R	A	R	R	S
3	93	0.08353211	R	A	A	S	R	A	R	R	S
4	94	0.07734522	R	R	A	S	R	A	R	R	S
5	64	0.05512233	R	R	A	R	S	R	R	A	S
6	52	0.05512233	R	R	A	R	S	A	R	A	S
7	62	0.05279333	R	R	A	S	S	R	R	A	S
8	63	0.05032836	R	A	A	R	S	R	R	A	S
9	61	0.05032836	R	A	A	S	S	R	R	A	S
10	50	0.04050067	R	R	A	S	S	A	R	A	S
11	239	0.02914362	R	A	A	R	R	A	R	S	R
12	287	0.02914362	R	A	A	R	R	R	R	A	R
13	285	0.02914362	R	A	A	S	R	R	R	A	R
14	275	0.02914362	R	A	A	R	R	A	R	A	R
15	263	0.02914362	R	A	A	R	R	S	R	A	R
16	251	0.02624536	R	A	A	R	R	R	R	S	R
17	237	0.02624536	R	A	A	S	R	A	R	S	R
18	273	0.02624536	R	A	A	S	R	A	R	A	R
19	16	0.02228159	R	R	A	R	S	A	R	S	S
20	261	0.02189156	R	A	A	S	R	S	R	A	R
21	249	0.01739729	R	A	A	S	R	R	R	S	R
22	227	0.01739729	R	A	A	R	R	S	R	S	R
23	225	0.01739729	R	A	A	S	R	S	R	S	R
24	311	0.01739729	R	A	A	R	R	A	R	R	R
25	299	0.01739729	R	A	A	R	R	S	R	R	R
26	28	0.00685636	R	R	A	R	S	R	R	S	S

Rank	Strategy	Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
1	28	0.333333333	R	R	A	R	S	R	R	S	S
2	93	0.083333333	R	A	A	S	R	A	R	R	S
3	94	0.083333333	R	R	A	S	R	A	R	R	S
4	95	0.083333333	R	A	A	R	R	A	R	R	S
5	96	0.083333333	R	R	A	R	R	A	R	R	S
6	225	0.010785634	R	A	A	S	R	S	R	S	R
7	227	0.010785634	R	A	A	R	R	S	R	S	R
8	237	0.033577442	R	A	A	S	R	A	R	S	R
9	239	0.033577442	R	A	A	R	R	A	R	S	R
10	249	0.010785634	R	A	A	S	R	R	R	S	R
11	251	0.033577442	R	A	A	R	R	R	R	S	R
12	261	0.010785634	R	A	A	S	R	S	R	A	R
13	263	0.033577442	R	A	A	R	R	S	R	A	R
14	273	0.033577442	R	A	A	S	R	A	R	A	R
15	275	0.033577442	R	A	A	R	R	A	R	A	R
16	285	0.033577442	R	A	A	S	R	R	R	A	R
17	287	0.033577442	R	A	A	R	R	R	R	A	R
18	299	0.010785634	R	A	A	R	R	S	R	R	R
19	311	0.010785634	R	A	A	R	R	A	R	R	R

Here, Strategy 28 placed first with a significantly larger share than any of the other strategies. This surge could be due to either an error or a stroke of evolutionary luck, as Strategy 28 was in the last position with the lowest share value of the prior results. Its proportion is equivalent to the shares of 93, 94, 95, and 96 combined. However, this does not seem to be a confounding variable in the simulation as it is the only strategy to be eliminated in the fourth round of the competing 19x19. Additionally note that this is the first observation of the mentioned Strategies 93, 94, 95, and 96 being equal constituents of the share population.

The 19x19 simulation executed in Round 5 results in the elimination of Strategy 28 and removal of 306 of the original 324 profiles, leaving us with the surviving 18. Here, we see some even convergence amongst these strategies. Once again, the highlighted Strategies 93, 94, 95, and 96 rank highest of the outstanding, with equal shares totaling to 50% of the population. The remaining 14 strategies share evenly the other 50%.

Rank	Strategy	Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
1	94	0.125	R	R	A	S	R	A	R	R	S
2	93	0.125	R	A	A	S	R	A	R	R	S
3	95	0.125	R	A	A	R	R	A	R	R	S
4	96	0.125	R	R	A	R	R	A	R	R	S
5	287	0.03571429	R	A	A	R	R	R	R	A	R
6	285	0.03571429	R	A	A	S	R	R	R	A	R
7	251	0.03571429	R	A	A	R	R	R	R	S	R
8	249	0.03571429	R	A	A	S	R	R	R	S	R
9	311	0.03571429	R	A	A	R	R	A	R	R	R
10	275	0.03571429	R	A	A	R	R	A	R	A	R
11	273	0.03571429	R	A	A	S	R	A	R	A	R
12	239	0.03571429	R	A	A	R	R	A	R	S	R
13	237	0.03571429	R	A	A	S	R	A	R	S	R
14	299	0.03571429	R	A	A	R	R	S	R	R	R
15	263	0.03571429	R	A	A	R	R	S	R	A	R
16	261	0.03571429	R	A	A	S	R	S	R	A	R
17	227	0.03571429	R	A	A	R	R	S	R	S	R
18	225	0.03571429	R	A	A	S	R	S	R	S	R

It is at this point in our testing where the tournament would technically conclude but the observations of the data consequent of the 18x18 lead us to conduct a final round with the top four strategy profiles. With half the population share dispersed evenly between the consistent top four, and the other half divided into shares of about 3.6% for the rest, the 14 strategies tying for last place seemed to be logical grounds to eliminate them and run another simulation with Strategies 93, 94, 95, and 96. To evaluate our conclusions of the tournament's replicator dynamics the aforementioned procedure was repeated for these strategies over a uniform distribution, disclosing a level convergence to the strategies:

Rank	Strategy	Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
1	96	0.25	R	R	A	R	R	A	R	R	S
2	95	0.25	R	A	A	R	R	A	R	R	S
3	93	0.25	R	A	A	S	R	A	R	R	S
4	94	0.25	R	R	A	S	R	A	R	R	S

4.1 Payoff Matrices

Payoff Matrix of 18 Fittest Strategy Profiles

Strategy	225	227	261	263	299	237	239	273	275	311	249	251	285	287	96	95	93	94
225	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
227	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
261	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
263	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
299	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
237	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
239	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
273	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
275	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
311	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
249	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
251	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
285	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
287	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0, 1)	(0, 1)	(0, 1)	(0, 1)
96	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)
95	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)
93	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)
94	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(1, 0)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)	(0.5, 0.5)

5 Conclusion

The tournament concluded with each carrying a quarter of the population share, observing the evolutionary implications of the interactions between many, slightly difference memes. From this analysis, simultaneity and awareness of prior moves becomes clear agents in the strategic tension studied in Standoff. Interestingly enough, the balance apparent in these findings suggests that these are evolutionarily stable strategies. When keeping in mind the player's perceived risk at each state, these contingency plans make intuitive sense. The inherent value of these strategies fluctuate due to their state dependency.

A tendency to shoot are can be considered strategically offensive, whereas a preference to armor implies players who are sensitive to risk and would be a part of a strategically defensive population. Reloading is strategically advancive, meaning such behavior confronts the greatest accumulative risk but permits players a greater range of choice and assurance in winning. This is especially true when considering larger player pools.

At state 1 Bullet vs. 1 Bullet, there is a clear evolutionary tendency to reload at this stage, armor and shoot become extinct as strategies and are no longer considered in the fittest 18 profiles. When comparing the actions of states 1 Bullet vs 2 Bullets and 2 Bullets vs. 1 Bullet at convergence, we see a growing proportion of armor at $s_t = (1, 2)$ and of reload at (2, 1). All four of the fittest strategies contain the most popular action at these states, respectively.

2 Bullets vs. 2 Bullets is the only state in which one of the three strategies, shoot, dominates under evolutionary analysis. This is due to armor being inferior to reload- which in this case ends the game- and this reduces the strategy space for this state to reload and shoot. Although either action may win the game, to reload still carries the risk of vulnerability while to shoot will either kill the other player or send both of them back to the null state. However only in the context of a two-player game is this assertion valid. For a player in a 2 Bullet state, reload (although riskier) may be a more viable strategy in the presence of more opponents.

Replicator dynamics allowed for the calculation of the equilibria for the two player game. Standoff carries a high probability of transitioning from a three to a two player state. Once the game has transitioned, Fx(a) represents a stage with the states null vs 1 bullet, while Fx(b) represents a stage with the states 1 bullet vs 2 bullets. In a stage with the states null vs 1 bullet, the action of armor is dominated. The best strategy for victory is to choose actions that protect the player while also focusing on reloading in order to acquire the bazooka. The resulting strategy consists of a strategy profile utilizing armor, reloading, and shooting when the best response is reliant on the states of the other players in each corresponding stage. In a stage where the other players are in the null state the best response is to reload.

In order for a player to survive they must consider armoring in stages where other players are in the 1 bullet or 2 bullets states because they can be shot. The player must take into account that the probability of being targeted increases with each additional bullet they acquire. In theory, one strategizing with a pattern of reloading and armoring would prove superior to a pattern consisting of shooting before attempting to acquire the bazooka. However, this strategy could compromise the player's success if the opponent(s) are strategizing in a similar manner. In a real-time simulation of the game, players are making decisions that they rationalize to be the best moves over an indefinite number of stages. This aspect of the game compounded with the possibility of an unlimited amount of players can produce an infinite amount of permutations of outcomes. Thus, without such limits on the player population or the number of stages, a retrospective analysis cannot be conducted nor can it fully comprehend the potential complexity of the game Standoff.

Rather, it is important to set parameters and reduce the game to one that permits a thorough analysis of the fast-paced, strategic contemplation and tension comprised within Standoff. Even when Standoff is condensed to a game with just two players, the various states that each player can encounter each other with creates complex strategy profiles comprised of pure and mixed Nash Equilibrium. In each stage the player must choose an action that is the best response to the opposing player's state in order to strategize to eliminate the other player and win the game.

Share	N vs. N	N vs. 1	N vs. 2	1 vs. N	1 vs. 1	1 vs. 2	2 vs. N	2 vs. 1	2 vs. 2
R	1	0.5	0	0.5	0.68421053	0.36842105	1	0.31578947	0.47368421
A	0	0.5	1	0.5	0	0.31578947	0	0.36842105	0
S	0	0	0	0	0.31578947	0.31578947	0	0.31578947	0.52631579

In the first stage where both players are in the null state the Pure NE is for both players to choose to reload. In a stage where both players are in the 1 bullet state the Mixed NE for both players is 0.309 Reload, 0.489 Armor, and 0.202 Shoot. In a stage where both players are in the 2 bullets state the Mixed NE for both players is 0.172 Reload, 0.414 Armor and 0.414 Shoot. In a stage where one player is in the 1 Bullet state and the other player is in the 2 Bullets state the Mixed NE for the 1 Bullet state player is 0.414 Reload, 0.293 Armor, and 0.293 Shoot. Whereas the Mixed NE for the 2 Bullets state player is 0.172 Reload, 0.586 Armor, and 0.242 Shoot. In a stage where one player is in the Null state and the other player is in the 1 bullet state the Mixed NE for the Null state player is to -0.631 Reload and 0.369 Armor. Whereas the Mixed NE for the 1 Bullet state player is to 0.631 Reload and 0.369 Shoot. In a stage where one player is in the Null state and the other player is in the 2 Bullets state the Pure NE is for the null player to Armor and the 2 Bullets state player to Reload.

Ratios calculated from the 19 x 19 strategy set

N vs N Reload

N vs 1 0.158 Reload and 0.841 Armor

N vs 2 Armor

1 vs N 0.579 Reload and 0.421 Armor

1 vs 1 0.947 Reload and 0.053 Shoot

1 vs 2 0.263 Reload, 0.474 Armor, and 0.263 Shoot

2 vs N Reload

2 vs 1 0.316 Reload, 0.316 Armor, and 0.368 Shoot

2 vs 2 0.737 Reload, 0.263 Shoot

Mixed Nash Equilibrium

N vs N Reload

N vs 1 0.631 Reload and 0.369 Armor

N vs 2 Armor

1 vs N 0.631 Reload and 0.369 Shoot

1 vs 1 0.309 Reload, 0.489 Armor, and 0.202 Shoot

1 vs 2 0.414 Reload, 0.293 Armor, and 0.293 Shoot

2 vs N   Reload

2 vs 1   0.172 Reload, 0.586 Armor, and 0.242 Shoot

2 vs 2   0.172 Reload, 0.414 Armor and 0.414 Shoot

We have been attempting to model the game as a finite state transition machine wherein the game moves from one state to another. We are fairly optimistic this will end up working better than the existing code we have, but sadly the time allotted after discovering this solution was not enough to properly complete its implementation. However, we plan to finish our work in its entirety so that the results are collinear with the mixed Nash Equilibrium solved through the matrix algebra. Also, we think that modeling it in this way gives us immense flexibility with respect to the extensibility to a three-player game since the transitions can be explicitly encoded into the game. We envision in the development of the new model to be an appropriate basis in the analysis of the three-player game for future study.

6 Appendix

6.1 Two Player Matrices

Bi-Matrices (2 Players)					
Bi-Matrix Diagram 1 Bullet vs 2 Bullets (Fx(b))					
			Player 2		
			R	*A*	*S*
	Player 1	*R*	(-1,1)	(0,0)	(-1,1)
		A	(-1,1)	(-0.414,0.414)	(0,0)
		S	(1,-1)	(-1,1)	(0,0)
State of Players after 1 Bullet vs 2 Bullets					
State of Player 1 (2 Bullets) or (1 Bullet) or (null)					
State of Player 2 (3 Bullets "Bazooka") or (2 Bullets) or (1 Bullet)					

Figure 4: Bi-matrices (2 Players)

Analysis: There is no pure Nash-Equilibrium for this stage because neither player has a strictly dominant strategy. However, a mixed NE exists. When no Pure NE exists probabilities are assigned to the various possible actions in order to derive the expected payoffs. A mixed NE is evaluated according to the expected payoff that is calculated. In this stage either player can choose to Reload, Armor or Shoot, but they can only choose one action. Thus, in terms of player 2 let p denote the probability that player 1 will choose reload, let q denote the probability that player 1 will choose armor and let (1-p-q) denote the probability that player 1 will choose to shoot. See below for the calculations giving the entries in yellow.

$$\pi_2(R) = 1 \cdot p + 1 \cdot q + -1 \cdot (1 - p - q) = 2 \cdot p + 2 \cdot q - 1$$

$$\pi_2(A) = 0 \cdot p + b \cdot q + 1 \cdot (1 - p - q) = b \cdot q + 1 - p - q$$

$$\pi_2(S) = 1 \cdot p + 0 \cdot q + 0 \cdot (1 - p - q) = p$$

$$\pi_2(R) = 2 \cdot p + 2 \cdot q - 1$$

$$\pi_2(A) = b \cdot q + 1 - p - q$$

$$\pi_2(S) = p$$

$$2 \cdot p + 2 \cdot q - 1 = b \cdot q + 1 - p - q = p = b$$

$$2 \cdot b + 2 \cdot q - 1 = b \cdot q + 1 - b - q = b$$

$$q = (2 \cdot b - 1)/(b - 1)$$

$$2 \cdot b + 2(2 \cdot b - 1)/(b - 1) - 1 = b$$

$$b^2 + 2 \cdot b - 1 = 0$$

$$b = \frac{-2 \pm \sqrt{8}}{2}$$

$$b = 0.414$$

In the mixed NE, player 1 has a higher chance of choosing reload, while player 2 has a higher chance of choosing armor. One possible explanation might be that when the game begins, player 2 has one more bullet than player 1, putting themselves at an advantage. Therefore, he or she could afford to play safe and choose armor with a higher probability. However, player 1 is at a disadvantage and knows if the opponent reloads, he or she will be defeated with Bazooka, so there is a greater incentive for them to take the risk and reload in this stage. The dominant payout for each action of each player is written in red.

1 Bullet vs. 2 Bullets:

Mixed NE: (0.414R+0.293A+0.293S, 0.172R+0.586A+0.242S)

Expected payoff: (-0.414, 0.414)

Bi-Matrix Diagram Null vs 1 Bullet (Fx(a))					
			Player 2		
			R	*A*	*S*
	Player 1	*R*	(-0.414,0.414)	(0,0)	(-1,1)
		A	(-1,1)	(-0.631,0.631)	(0,0)
State of Players after Null vs 1 Bullet					
State of Player 1 (1 Bullet) or (null)					
State of Player 2 (2 Bullets) or (1 Bullet) or (null)					

Figure 5: Bi-matrices (2 Players)

$$\pi_1(R) = -b \cdot P + -1 \cdot (1 - P) = -b \cdot P - 1 + P$$

$$\pi_1(A) = -1 \cdot P + 0 \cdot (1 - P) = -P$$

$$-b \cdot P - 1 + P = -P$$

$$P = \frac{1}{2-b} = \frac{1}{2-0.414} = 0.631$$

$$\text{Payoff at NE is}(-0.631, 0.631) = (-a, a)$$

$$a = 0.631$$

Null vs. 1 Bullet

For player 2, A will not be chosen at any time. So the payoff matrix is trimmed down to a 2 strategy by 2 strategy matrix with player 1 chooses from {R, A} and player 2 chooses from {R, S}.

No pure NE exists.

Mixed NE: {(-0.631R+0.369A, 0.631R+0.369S)} Expected payoff: (-0.631, 0.631)

Analysis: If player 1 is in the null state and player 2 has 1 bullet the stage will result in a mixed Nash-Equilibrium. Player 1 has a probability of 0.631 to reload and 0.369 to arm. While player 2, there is a 0.631 chance he or she will choose to reload and 0.369 chance he or she will shoot.

Bi-Matrix Diagram Null vs Null					
			Player 2		
			R	*A*	
	Player 1	*R*	(0,0)	(0.631,-0.631)	
		A	(-0.631,0.631)	(0,0)	
State of Players after Null vs Null					
State of Player 1 (1 Bullet) or (null)					
State of Player 2 (1 Bullet) or (null)					

Figure 6: Bi-matrices (2 Players)

Null vs. Null

Pure NE: {(R, R)}

Payoff: (0, 0)

Analysis: When both players are in the null state, the best response is for them to both reload. Armor will never be chosen, because neither player cannot be shot by the other player. Reloading will allow them to shoot in the next stage, while armoring will put them at a disadvantage. Thus, armoring is strictly dominated and the best response for both players is to reload.

Bi-Matrix Diagram Null vs 2 Bullets					
			Player 2		
			R	*A*	*S*
	Player 1	*R*	(-1,1)	(-0.414,0.414)	(-1,1)
		A	(-1,1)	(-1,1)	(-0.631,0.631)
State of Players after Null vs 2 Bullets					
State of Player 1 (1 Bullet) or (null)					
State of Player 2 (Bazooka) or (2 Bullets) or (1 Bullet)					

Figure 7: Bi-matrices (2 Players)

Null vs. 2 Bullets

Pure NE: {(A, R)}

Payoff: (-1, 1)

Analysis: Pure Nash-Equilibria exist for the stage where player 1 is in the null state and player 2 has 2 bullets. Player 1's best response is to armor, while player 2's best response is to reload. Player 1 is not able to shoot player 2, so player 2 will not take the risk of shooting in this stage in case player 1 chooses armor. Rather, the best response is to reload in order to acquire the Bazooka and proceed to win the game in the following stage. Thus, shooting and armor are dominated actions for player 2. While for player 1, armor strictly dominates reload. So he or she will choose to armor anyhow.

$$\pi_1(R) = 0 \cdot P + 0.414 \cdot Q + -1 \cdot (1 - P - Q) = 0.414 \cdot Q + P + Q = -1 + P + 1.414 \cdot Q$$

$$\pi_1(A) = -0.414 \cdot P + 0 \cdot Q + 0.631 \cdot (1 - P - Q) = 0.217 \cdot P - 0.631 \cdot Q + 0.631$$

$$\pi_1(S) = 1 \cdot P + -0.631 \cdot Q + 0 \cdot (1 - P - Q) = P - 0.631 \cdot Q$$

$$\pi_1(R) = -1 + P + 1.414 \cdot Q$$

Bi-Matrix Diagram 1 Bullet vs 1 Bullet					
			Player 2		
			R	*A*	*S*
	Player 1	*R*	(0,0)	(0.414,-0.414)	(-1,1)
		A	(-0.414,0.414)	(0,0)	(0.631,-0.631)
		S	(1,-1)	(-0.631,0.631)	(0,0)
State of Players after 1 Bullet vs 1 Bullet					
State of Player 1 (2 Bullets) or (1 Bullet) or (null)					

Figure 8: Bi-matrices (2 Players)

$$\pi_1(A) = 0.217 \cdot P - 0.631 \cdot Q + 0.631$$

$$\pi_1(S) = P - 0.631 \cdot Q$$

$$-1 + P + 1.414 \cdot Q = 0.217 \cdot P - 0.631 \cdot Q + 0.631 = P - 0.631 \cdot Q$$

$$Q = 0.489$$

$$-1 + P + 1.414(0.489) = 0$$

$$P = 0.309$$

$$1 - P - Q = 0$$

$$1 - 0.309 - 0.489 = 0.202$$

1 Bullet vs. 1 Bullet

Mixed NE: {(0.309R+0.489A+0.202S, 0.309R+0.489A+0.202S)}

Expected payoff: (0, 0)

Analysis: There is no pure Nash-Equilibrium for this stage because no strictly dominated strategies exist. However, a mixed NE exists that results in a probability of 0.309 for choosing reload, 0.489 for choosing armor, and 0.202 for choosing shoot for both players. Since this is a zero-sum game, the expected payoff would be 0 for both players.

$$\pi_1(R) = 0 \cdot P + 1 \cdot Q + -1 \cdot (1 - P - Q) = -1 + P + 2 \cdot Q$$

$$\pi_1(A) = -1 \cdot P + 0 \cdot Q + 0.414 \cdot (1 - P - Q) = -1.414 \cdot P - 0.414 \cdot Q + 0.414$$

$$\pi_1(S) = 1 \cdot P - 0.414 \cdot Q + 0 \cdot (1 - P - Q) = P - 0.414 \cdot Q$$

$$\pi_1(R) = -1 + P + 2 \cdot Q$$

Bi-Matrix Diagram 2 Bullets vs 2 Bullets					
			Player 2		
			R	*A*	*S*
	Player 1	*R*	(0,0)	(1,-1)	(-1,1)
		A	(-1,1)	(0,0)	(0.414,-0.414)
		S	(1,-1)	(-0.414,0.414)	(0,0)
State of Players after 2 Bullets vs 2 Bullets					
State of Player 1 (Bazooka) or (2 Bullets) or (1 Bullet) or (null)					
State of Player 2 (Bazooka) or (2 Bullets) or (1 Bullet) or (null)					

Figure 9: Bi-matrices (2 Players)

$$\pi_1(A) = -1.414 \cdot P - 0.414 \cdot Q + 0.414$$

$$\pi_1(S) = P - 0.414 \cdot Q$$

$$1 + P + 2 \cdot Q = -1.414 \cdot P - 0.414 \cdot Q + 0.414 = P - 0.414 \cdot Q$$

$$1 + P + 2 \cdot Q = P - 0.414 \cdot Q$$

$$Q = 0.414$$

$$-1 + P + 2 \cdot (0.414) = 0$$

$$P = 0.172$$

$$1 - P - Q = 0$$

$$1 - 0.172 - 0.414 = 0.414$$

2 Bullet vs. 2 Bullets

Mixed NE: {(0.172R+0.414A+0.414S, 0.172R+0.414A+0.414S)}

Expected payoff: (0, 0)

Analysis: There is no pure Nash-Equilibrium for this stage because no dominated strategies exist. However, a mixed NE exist that result in a probability of 0.414 for choosing armor or shoot, and 0.172 for choosing reload for both players. Since this is a zero-sum game, the expected payoff would be 0 for them both. There is a relatively small probability of choosing to reload, for both players know the opponent has 2 bullets, thus it is risky to reload and leave themselves vulnerable. Even without reloading, they still retain their amount of bullets, therefore they are not too willing to take the risk of losing to get another bullet. Both players know that reloading to acquire the bazooka is an action that will result in both players entering the next stage in the null state. This payout calculates that the players will be indifferent because they will both be in the null state in the following stage.

6.2 Python Code - Standoff Game

```
import sys
import csv
import itertools
import numpy as np

SHOT_FIRED = False

class player(object):

	def __init__(self):
		# stores the state before transition
		# upon shooting from bullet1 => old = b1
		# n - null, b1 - bullet1, b2 - bullet2, bz - bazooka

		self.old = 'n'
		self.null = True
		self.bullet1 = False
		self.bullet2 = False
		self.bazooka = False
		self.win = False
		self.dead = False

	def reload(self):
		if self.null:
			self.null = False
			self.bullet1 = True
			self.old = 'n'
		elif self.bullet1:
			self.bullet1 = False
			self.bullet2 = True
			self.old = 'b1'
		elif self.bullet2:
			self.bullet2 = False
			self.bazooka = True
			self.old = 'b2'

	def fire_bullet(self):
		if self.bullet1:
			self.old = 'b1'
			self.bullet1 = False
			self.null = True
			global SHOT_FIRED
```

```
                        SHOT_FIRED = not(SHOT_FIRED)
                elif self.bullet2:
                        self.old = 'b2'
                        self.bullet2 = False
                        self.bullet1 = True
                        global SHOT_FIRED
                        SHOT_FIRED= not(SHOT_FIRED)
                else:
                        return False

        def fire_bazooka(self):
                if self.bazooka:
                        self.old = 'b2'
                        # self.bazooka = False
                        self.null = True

        def take_bullet(self):
                self.dead = True

        def armor(self):
                if SHOT_FIRED:
                        SHOT_FIRED = not(SHOT_FIRED)
                else:
                        print "Armored for no reason!"
                self.old = self.old


actionDict = {'R':1, 'S':2, 'A':3}

def extract_strategies(file):
        strategies = []
        with open(file) as f:
                reader = csv.DictReader(f)
                for row in reader:
                        strategies.append(row)
        return strategies

def strategy_map(strategy, player1, player2, playerTurn):
        if playerTurn == 1:
                if player1.null and player2.null:
                        # player1.reload()
                        return actionDict[strategy['N vs N']]
                if player1.null and player2.bullet1:
                        # player1.reload()
                        return actionDict[strategy['N vs 1']]
                if player1.null and player2.bullet2:
                        # player1.reload()
                        return actionDict[strategy['N vs 2']]
                if player1.bullet1 and player2.null:
                        # player1.reload()
                        return actionDict[strategy['1 vs N']]
```

```
                if player1.bullet1 and player2.bullet1:
                        # player1.fire_bullet()
                        return actionDict[strategy['1 vs 1']]
                if player1.bullet1 and player2.bullet2:
                        # player1.fire_bullet()
                        return actionDict[strategy['1 vs 2']]
                if player1.bullet2 and player2.null:
                        # player1.fire_bullet()
                        return actionDict[strategy['2 vs N']]
                if player1.bullet2 and player2.bullet1:
                        # player1.reload()
                        return actionDict[strategy['2 vs 1']]
                if player1.bullet2 and player2.bullet2:
                        # player1.reload()
                        return actionDict[strategy['2 vs 2']]
        if playerTurn == 2:
                if player1.null and player2.old=='n':
                        # player1.reload()
                        return actionDict[strategy['N vs N']]
                if player1.null and player2.old=='b1':
                        # player1.reload()
                        return actionDict[strategy['N vs 1']]
                if player1.null and player2.old=='b2':
                        # player1.reload()
                        return actionDict[strategy['N vs 2']]
                if player1.bullet1 and player2.old=='n':
                        # player1.reload()
                        return actionDict[strategy['1 vs N']]
                if player1.bullet1 and player2.old=='b1':
                        # player1.fire_bullet()
                        return actionDict[strategy['1 vs 1']]
                if player1.bullet1 and player2.old=='b2':
                        # player1.fire_bullet()
                        return actionDict[strategy['1 vs 2']]
                if player1.bullet2 and player2.old=='n':
                        # player1.fire_bullet()
                        return actionDict[strategy['2 vs N']]
                if player1.bullet2 and player2.old=='b1':
                        # player1.reload()
                        return actionDict[strategy['2 vs 1']]
                if player1.bullet2 and player2.old=='b2':
                        # player1.reload()
                        return actionDict[strategy['2 vs 2']]


def main():


        strategies = extract_strategies("Complete_Strategy_Set.csv")
        games = list(itertools.product(strategies, strategies))
        matrix = []
```

```
j = 0
array_games = []
for game in games:
    # print game
    j = j+1
    gameOver = False
    playerTurn = [1, 2]
    i = 0
    shotFired = False
    bazookaFired = False
    p1 = player()
    p2 = player()
    inf_check = 0

    while not gameOver:
        inf_check += 1
        if not p1.dead and not p2.dead:
            # print "Player {} choose: (1) Reload, (2) Shoot,
    (3) Armor (4) Bazooka".format(str(playerTurn[i]))
            # choice = input("Choice:\n")

            if playerTurn[i] == 1:
                choice = strategy_map(game[0],p1,p2,
      playerTurn[i])
            else:
            # //////////////////////
            # (p2,p1) now because you're playing wrt p2
                choice = strategy_map(game[1],p2,p1,
      playerTurn[i])
            # //////////////////////

            # print "Player: " + str(playerTurn[i]) +
    " Choice: " + str(choice)
            # sys.exit()


            if playerTurn[i] == 1:  # player 1's turn
                if choice == 1:
                    if shotFired:
                        p1.take_bullet()
                    else:
                        p1.reload()
                        if bazookaFired and
          p2.bazooka == False:
                            p2.dead = True
                elif choice == 2 and p1.bullet1
      == False and p1.bullet2 == False:
                    # print "You have no bullets!
        Choose again!"
                    continue
                elif choice == 2:
```

```
                p1.fire_bullet()
                shotFired = not(shotFired)
            elif choice == 3:
                if shotFired:
                    # print "Shielded"
                    shotFired = not(shotFired)
                elif bazookaFired and
    p2.bazooka == False:
                    p2.dead = True
                # else:
                    # print "Shielded
      for no reason!"

            if choice == 1 and shotFired:
  # player 1 hasn't fired/shielded
                p1.take_bullet()

            if p1.bazooka and not shotFired:
                p1.fire_bazooka()
                bazookaFired = not(bazookaFired)

        elif playerTurn[i] == 2:
# player 2's turn
            if choice == 1:
                if shotFired:
                    p2.take_bullet()
                else:
                    p2.reload()
                    if bazookaFired and
      p2.bazooka == False:
                        p2.dead = True

            elif choice == 2 and p2.bullet1
  == False and p2.bullet2 == False:
                # print "You have no bullets!
    Choose again!"
                continue
            elif choice == 2:
                p2.fire_bullet()
                if p1.bazooka and shotFired:
                    p1.dead = True

                shotFired = not(shotFired)
            elif choice == 3:
                if shotFired:
                    # print "Shielded"
                    shotFired = not(shotFired)
                elif bazookaFired and
    p2.bazooka == False:
                    p2.dead = True
                # else:
```

```
                                        # print "Shielded
            for no reason!"

                                elif choice == 1 and shotFired:
        # player 2 hasn't fired/shielded
                                        p2.take_bullet()

                                elif p1.bazooka and p2.bazooka
        and not shotFired and not
        bazookaFired:
                                        p2.fire_bazooka()
                                        inf_check = 100

                                elif p1.bazooka and p2.bazooka
        and bazookaFired==False:
                                        inf_check = 100

                                elif p1.bazooka and not
        p2.bazooka:
                                        p2.dead = True

                                elif not p1.bazooka and
        p2.bazooka:
                                        p1.dead = True

                                elif p1.dead == False and
        bazookaFired and p2.bazooka:
                                        inf_check = 100


                        i = (i+1) % 2

                # if p1.dead and p2.dead or p1.win
and p2.win:
                        # continue

                if p1.dead or p2.win:
                        # print game[1]["Index"] + " wins!"
                        # array_games.append((-1,1))
                        array_games.append(0)
                        gameOver = True
                elif p2.dead or p1.win:
                        # print game[0]["Index"] + " wins!"
                        # array_games.append((1,-1))
                        array_games.append(1)
                        gameOver = True

                if inf_check > 100:
                        # print "Draw!!"
                        # array_games.append((0,0))
                        array_games.append(0.5)
```

```
                                        gameOver = True

                        if j%324 == 0:
                                matrix.append(array_games)
                                array_games = []


        with open("out.csv", "w") as f:
                writer=csv.writer(f)
                for row in matrix:
                        writer.writerow(row)

        matrix2 = []
        for row in matrix:
                row2 = []
                for column in row:
                        if column == 1:
                                row2.append((1,-1))
                        elif column == 0:
                                row2.append((-1,1))
                        else:
                                row2.append((0,0))
                matrix2.append(row2)

        with open("out2.csv", "w") as f:
                writer=csv.writer(f)
                for row in matrix2:
                        writer.writerow(row)

if __name__ == '__main__':
        main()
```

6.3 Python Code - Replicator Dynamics

```
import csv
import sys
import operator

payoff_matrix = []
shares = [1.0/324 for i in range(324)]

with open("out.csv", "r") as f:
        reader = csv.reader(f)
        for row in reader:
                row = [float(ele) for ele in row]
                payoff_matrix.append(row)

for i in range(1000):
        print "{}".format(i+1)
        i += 1
        shares_matrix = []
        for row in payoff_matrix:
```

```
            temp = []
            for payoff,share in zip(row,shares):
                    temp.append(payoff \cdot share)
            shares_matrix.append(temp)

        fitness_vector = [sum(row) for row in shares_matrix]

        average_fitness = sum([share \cdot
    w for share,w in zip(shares,fitness_vector)])

        with open("rep.csv", "a") as f:
                writer = csv.writer(f)
                temprow = [str(ele) for ele in shares] +
        [""] + [str(ele) for ele in fitness_vector]
        + [""] + [str(average_fitness)]
                writer.writerow(temprow)

        shares = [(s \cdot w)/average_fitness for s,w in
    zip(shares,fitness_vector)]

d = {}
for i in range(324):
        d[i] = shares[i]

with open("rep.csv", "a") as f:
        f.write("\n\n")
        f.write("Sorted:")
        writer = csv.writer(f)
        writer.writerow(sorted(d.items(),
    key=operator.itemgetter(1)))

print sorted(d.items(), key=operator.itemgetter(1))
```

6.4 R Code - To Generate Strategies

This code was written to compute all 324 permutations of the non-dominated strategies per stage.

```
setwd("/Users/William/Dropbox/Standoff Game/")
getwd()

v1 <- c("R")
v2 <- c("A", "R")
v3 <- c("A")
v4 <- c("S", "R")
v5 <- c("S", "A", "R")
v6 <- c("S", "A", "R")
v7 <- c("R")
v8 <- c("S", "A", "R")
v9 <- c("S", "A", "R")

m9 <- matrix(data = v9, nrow = 3, ncol = 1, byrow = FALSE, dimnames = NULL)

join_vector <- function(add_V, to_M) {
```

```
  counter <- 1
  r <- matrix(data = NA, nrow = length(add_V) \cdot
  dim(to_M)[1], ncol = 1+dim(to_M)[2], byrow = FALSE, dimnames = NULL)
  for (i in 1:dim(to_M)[1]) {
    for (j in 1:length(add_V)) {
      r[counter, ] <- c(add_V[j], to_M[i, ])
      print(add_V[j])
      counter <- counter + 1
    }
  }
  return(r)
}

m8 <- join_vector(v8, m9)
m7 <- join_vector(v7, m8)
m6 <- join_vector(v6, m7)
m5 <- join_vector(v5, m6)
m4 <- join_vector(v4, m5)
m3 <- join_vector(v3, m4)
m2 <- join_vector(v2, m3)
m1 <- join_vector(v1, m2)

write.table(x = m1, file = "/Users/William/Dropbox/Standoff Game/complete.csv")
```

References

[Axe06] Robert M Axelrod. *The Evolution of Cooperation.* Basic Books, 2006.

[BK98] Steven J Brams and D Marc Kilgour. Backward induction is not robust: The parity problem and the uncertainty problem. *Theory and Decision*, 45(3):263–289, 1998.

[Fri91] Daniel Friedman. Evolutionary games in economics. *Econometrica: Journal of the Econometric Society*, pages 637–666, 1991.

[FS16] Daniel Friedman and Barry Sinervo. *Evolutionary Games in Natural, Social, and Virtual Worlds.* Oxford University Press, 2016.

[Har09] Joseph Harrington. *Games, Strategies and Decision Making.* Macmillan, 2009.

[KB97] D. Marc Kilgour and Steven J. Brams. The truel. *Mathematics Magazine 70*, pages 315–326, 1997.

[Kil72] D. Marc Kilgour. The simultaneous truel. *International Journal of Game Theory 1*, pages 229–242, 1972.

[Kil73] D. Marc Kilgour. *Duels, Truels and n-uels.* PhD thesis, University of Toronto, 1973.