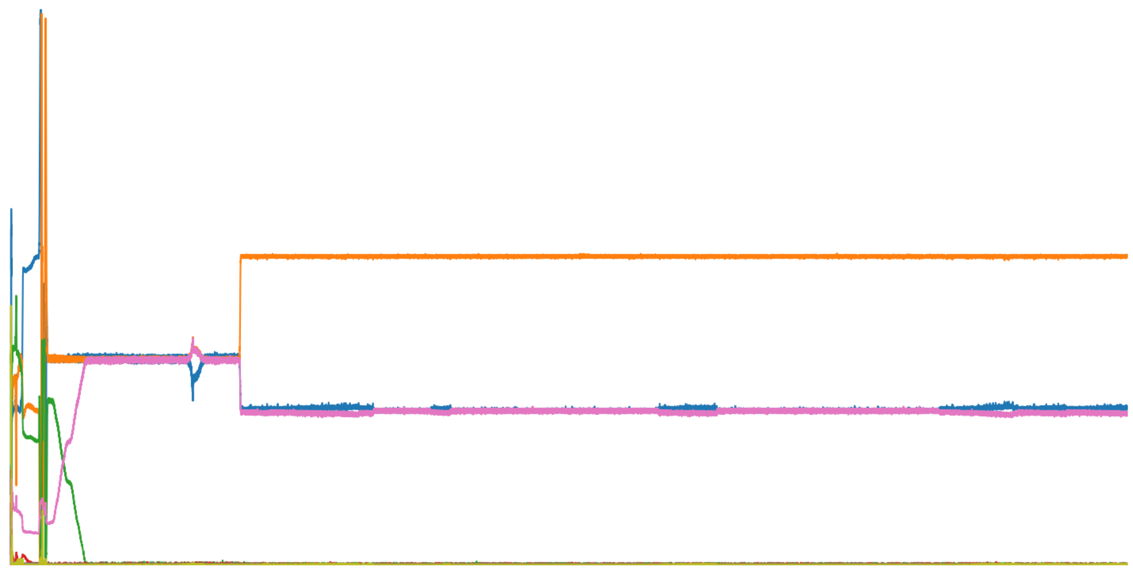
CHALMERS
UNIVERSITY OF TECHNOLOGY



Evolutionary voting games

Master's thesis in Complex Adaptive Systems

CARL FREDRIKSSON

Department of Space, Earth and Environment
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

MASTER'S THESIS 2018

Evolutionary voting games

CARL FREDRIKSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Space, Earth and Environment
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Evolutionary voting games
CARL FREDRIKSSON

Supervisor and Examiner: Kristian Lindgren, Department of Space, Earth and Environment, Physical Resource Theory

Master's Thesis 2018
Department of Space, Earth and Environment
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A plot of winning coalitions over time in an evolutionary simulation of a voting game. Each color represents a coalition. The y-axis is the proportion of games won. The x-axis is the amount of evolutionary iterations that has passed.

Typeset in LaTeX
Gothenburg, Sweden 2018

Evolutionary voting games
CARL FREDRIKSSON
Department of Space, Earth and Environment
Chalmers University of Technology

Abstract

A voting game is a cooperative game that can be used as a model for political elections. In such a model the players are political parties that have different amount of votes, and the goal is to form a coalition that has a majority of all votes. The winning coalition receives a budget to divide freely between its members, and the players' payoff is the amount of the budget they get. Players want to know how much payoff they will receive before agreeing to join a coalition. This could lead to an endless bargaining process, which is why I have formalized a simple process that limits the actions of players. The goal of this thesis is to model an evolutionary development of strategies for a voting game with a limited bargaining process, and to analyze the results. Having a limited bargaining process makes it possible to also model the game as a non-cooperative game, which allows the results to be analyzed using solution concepts from both cooperative and non-cooperative game theory.

In the evolutionary development, long term stability of strategies is often observed. In some of those cases a Nash equilibrium has formed, and we can predict that the stability will last indefinitely. In the other cases there exists at least one better strategy that eventually would be found and break the stability. When averaging several resulting top strategies for the different player positions, we observe a vector that resembles the Shapley values for those positions.

Keywords: Game theory, Cooperative game theory, Evolutionary game theory, Voting game, Replicator dynamics

Acknowledgements

I would like to thank my examiner and supervisor Kristian Lindgren. His guidance have been invaluable, especially with regards to modelling the evolutionary strategy development.

Carl Fredriksson, Gothenburg, June 2018

Contents

1

Introduction

1.1 Background

Game theory has two major sub fields called non-cooperative game theory and cooperative game theory. The most studied sub field is non-cooperative game theory, where one considers the individual actions of players who has no ability to form binding contracts with other players. Cooperation can still occur, but has to be a product of the players' self interest. For example, in the infinitely iterated Prisoner's dilemma game, it is mutually beneficial for the two players to play strategies that will result in both cooperating most of the time. Non-cooperative game theory focuses on predicting the actions and resulting payoffs for individual players, as well as analyzing Nash equilibria [3].

In cooperative game theory players can form binding contracts, which results in coalitions of players. Coalitions will get payoffs depending on what members they contain. The payoff can be freely divided between the members of the coalition. Players normally want to know about the division of payoff before agreeing to enter a coalition. The main objects of study in cooperative game theory are the formation of coalitions and the possible payoff divisions. Instead of analyzing Nash equilibria, other solution concepts such as the core and the Shapley value are studied. There are many types of cooperative games, one of which are voting games. Voting games can be used to model political parties that want to form a coalition that receives the majority of the votes in an election. The winning coalition get to decide on how to divide the a budget of money, and the other coalitions get nothing. The objective for each player (political party) is to get as much of the payoff (budget) as possible. Each party has a different amount of votes, and it is natural to assume that a party that brings more votes to the coalition should get a bigger part of the payoff. The notion that players who bring more value to coalition is one of the driving principles for the cooperative solution concepts mentioned above [1],[3].

Evolutionary game theory is an application of game theory, where strategies are treated as individuals in a population governed by dynamics inspired by Darwinian evolution. It originated in 1973 when John Maynard Smith and George R. Price used computer simulations to explain conflicts between animals [8]. Just like normal game theory, it can be divided into cooperational and non-cooperational variants. Evolutionary non-cooperative game theory has been studied in many different contexts, for example in [4] and [2]. Evolutionary cooperative game theory is more rare, but has been studied in [7].

The aim of this thesis is to study evolutionary dynamics of strategies for voting games with a simple bargaining process to formalize how coalitions are formed. The formalization allows us to study the games through the lens of both cooperative and non-cooperative evolutionary game theory.

1.2 Objectives

The primary goal of this thesis is to identify and characterize successful strategies in voting games, by using an evolutionary approach in combination with a simple bargaining process. I will compare results from computer simulations with theoretical solution concepts from both cooperative and non-cooperative game theory. The coalitions that are formed and how the payoff is divided will be compared with cooperative solution concepts. The proposals that are given and how the players are voting will be compared with non-cooperative solution concepts.

1.3 Method

By formalizing a coalition forming procedure, a cooperative game can be turned into a non-cooperative game. In this thesis I use a very simple bargaining process where players take turns proposing coalitions with payoff splits, and the proposed players get to vote yes or no on the proposal. If all proposed players vote yes the coalition is formed, the payoff is split according to the proposed split, and the game is over. Each player has a finite amount of individual actions she can perform at any given point, such as what to propose when it is your turn or what to vote when proposed. Thus the game can be modeled as a non-cooperative game. We can still analyze the game from the perspective of cooperative game theory, to see what coalitions form and how the payoff is divided. As mentioned above I will consider both perspectives.

I have created a simulation environment using the Python programming language and scientific programming packages such as NumPy. The simulation environment lets the user run a specified amount of evolutionary simulations. Each run contains a strategy population for each position in the voting game. Each position could model a political party as in the example above, and has a corresponding ”voting power”. Voting power would model the amount of voters that party brings to a coalition. Each run will run for a specified number of iterations where the positions' strategy populations are governed by evolutionary dynamics. The evolutionary dynamics has three major components: Replicator dynamics inspired by the paper from Eriksson and Lindgren about the multi-person Prisoner's dilemma [2], removal of dying strategies, and strategy mutations. After a simulation run data is saved to be compared later with theory.

The simulation is limited in many ways. I only consider one very simple bargaining process and the strategy space could be made more complex. With this in mind, a possible avenue for future research could be to extend the simulation to be able to model more complex behaviour.

2

Theory

2.1 Game theory

I will start by explaining the general concept of game theory. I will then give a brief introduction to non-cooperative game theory, followed by cooperative game theory and evolutionary game theory. Most of the content of this chapter is heavily inspired by Leyton-Brown and Shoham's excellent book "Essentials of Game Theory: A Concise, Multidisciplinary Introduction" [3].

"Game theory is the mathematical study of interaction among independent, self-interested agents" [3]. Game theory was initially developed for application in economics, but has now been used in a wide variety of fields, including biology, political science, and many more. When applying game theory to a real world problem, agents are modeled as players in a game representing the real world scenario. There are different ways to represent games, which will be discussed later.

Agents are often assumed to be rational profit-maximizers, which means that they choose actions for the sole purpose of maximizing some quantity, such as money, for themselves. This rationality assumption can be dubious when trying to model human behaviour, since evidence from cognitive psychology, anthropology, evolutionary biology, and neurology have shown that consumers rarely are completely rational [5]. Even though this might be the case, game theory can still be used to model a wide range of real world applications. The rationality assumption can be approximate or we can study systems from a higher level of abstraction where the rationality assumption is more accurate, for example on the level of companies in an economy.

2.2 Non-cooperative game theory

The non-cooperative sub-field of game theory models games where players can not form binding coalitions with other players. Each player has a corresponding payoff function which takes the state of the game and maps to a real valued payoff. The object for each player is to maximize that payoff. The name non-cooperative can be misleading since cooperation can still occur, but it has to be an emergent feature of the players' self-interest, rather than a built in feature of the game studied.

2.2.1 Normal-form games

The most fundamental ways to represent a game is in normal form. There are other representations, but most of them can be reduced to normal form.

The definition of a finite, n-person normal-form game is a tuple (N, A, p), where

- N is a finite set of n players, indexed by i
- $A = A_1 \times \cdots \times A_n$, where A_i is a finite set of actions available to player i. Each vector $a = (a_1, \ldots, a_n) \in A$ is called an action profile
- $p = (p_1, \ldots, p_n)$ where $p_i : A \mapsto \mathbb{R}$ is a real valued payoff function for player i

2.2.2 Prisoner's dilemma

One of the most iconic normal form games studied in game theory is the Prisoner's dilemma game. I will use this as an example to illustrate what a normal-form game can look like. One of the ways to represent a normal-form game is by a n dimensional matrix. Usually in a two player game, each row corresponds to an action that player 1 can take, and each column corresponds to an action that player 2 can take. The cells show the payoff vector for that action profile, with the first value being the payoff for player 1 and the second for player 2. Below is a matrix representation of the Prisoner's dilemma game, given that $c > a > d > b$.

Table 2.1: Matrix representation of the Prisoner's dilemma game

	C	D
C	a,a	b,c
D	c,b	d,d

With some values for a, b, c, d:

Table 2.2: Matrix representation of the Prisoner's dilemma game

	C	D
C	3,3	0,5
D	5,0	1,1

The "C" action stands for cooperate, and the "D" action stands for defect. If two players play the game once, it is easily seen that regardless of what the other player chooses, one should always play the defect action. If the other player cooperates, you will gain a payoff of 5 which of course is greater than the 3 you would gain by playing cooperate as well. If the other player defects, you will gain a payoff of 1 which of course is greater than the 0 you would gain by cooperating.

The game becomes more interesting if played for a number of rounds while accumulating payoff. It is obviously mutually beneficial for the players to both choose cooperate rather than both choosing defect, and when the game is played for more than one round there is time for players to establish cooperation. Even if players establish cooperation, it is very fragile since when a player expect another of cooperating, he can of course be tempted to switch to defect in order to exploit the other. These ideas will be explored in more detail in later sections.

2.2.3 Strategies

So far we have only discussed the actions that players can take. Strategies for how to choose actions can be very simple or more complex, depending on the game. In a game played once, the most simple strategy is just deciding on an action and playing it. We call this kind of strategy a pure strategy. A strategy can also be probabilistic, so that each available action has a given probability of being selected. We call this kind of strategy a mixed strategy. A pure strategy is a special case of a mixed strategy with the probability of choosing some action set to 100%. An example of a normal-form game where it is intuitive to play a mixed strategy is the classic rock-paper-scissors game. The game can be summarized by: rock beats scissors, scissors beat paper, and paper beats rock. All actions draw vs themselves. The game can be represented in matrix form as follows:

Table 2.3: Matrix representation of the Rock-paper-scissors game

	Rock	Paper	Scissors
Rock	0,0	-1,1	1,-1
Paper	1,-1	0,0	-1,1
Scissors	-1,1	1,-1	0,0

If more than one round is played a pure strategy is easily exploited, and most people usually resort to using a mixed strategy (even if they do not have any exact probabilities in mind). In fact all strategies that does not play each action with the same probability 1/3 can be exploited. For example if an opponent plays rock with probability 2/3 and the other actions with probability 1/6 each, then you can play paper as a pure strategy and win 2/3 of the time, lose 1/6 of the time, and draw 1/6 of the time.

In an iterated game strategies can be more complex than in one round games. In particular strategies can be a function of the action history for all players. A simple example of such as a strategy is the "tit-for-tat" strategy in the iterated Prisoner's dilemma game. When playing this strategy the player starts off by cooperating, and then continues by mirroring the opponent's last action. This strategy attempts to cooperate with the opponent while being quick to change if the opponent is trying to exploit by defecting.

The set of all strategies available for player i is denoted S_i. A vector of chosen strategies for the n players $s = (s_1, \ldots, s_n)$ is usually called a strategy profile.

2.2.4 Nash equilibrium

The most influential solution concept in non-cooperative game theory is the Nash equilibrium. Before we can properly define it I will introduce the concept of a best response strategy. A best response strategy is a strategy that maximizes the expected payoff for a player, given that the strategies for the other players are known. More formally, if s_{-i} is a strategy profile that contains strategies for all players except player i, then a best response strategy for player i to s_{-i} is a strategy $s_i^* \in S_i$ that satisfies:

$$p_i(s_i^*, s_{-i}) \geq p_i(s_i, s_{-i}), \quad \forall s_i \in S_i \tag{2.1}$$

where $p_i(s_i, s_{-i})$ is the expected payoff for strategy s_i playing versus the strategy profile s_{-i}.

A Nash equilibrium is a strategy profile where no player would want to change his action if he knew what strategies the others were playing. More formally $s = (s_1, \ldots, s_n)$ is a Nash equilibrium if, for all players i, s_i is a best response strategy to s_{-i}. A Nash equilibrium is called strict if all best response strategies are unique, and weak if this is not the case. Strict Nash equilibria are only possible when all strategies in the strategy profile are pure strategies.

As an example I will start with the one round Prisoner's dilemma game. In this game the only Nash equilibrium is $s = (D, D)$. We can confirm that this is a Nash equilibrium by freezing player 2's action as defect, and checking the utility for the available actions for player 1. We only have two available actions to check and since $s = (C, D)$ will net a payoff of 0 to player 1, while $s = (D, D)$ will give him a payoff of 1 we can conclude that defect is the best response to defect. Since the pure strategy profile space is very small we can try the other pure strategy profiles and rule them out. In all other pure profiles one of the players is not playing his best response strategy. There are no non-pure Nash equilibria since defect is the best response to both cooperate and defect.

In the iterated Prisoner's dilemma game it becomes more interesting. If the number of rounds is known and finite we still have only one Nash equilibrium characterized by $s = (D, D)$ in every round. This can be seen by a technique called backward induction. Backward induction works by looking at the best responses for the last round and working backwards. It is clearly mutually beneficial for both players to cooperate the majority of the rounds, rather than both defecting. The problem is that since the last round can be seen as a one round Prisoner's dilemma game, it is a better strategy to do some cooperation with your opponent except in the last round. But since it is now known that both should play defect in the last round, we can treat the next to last round the same. Stepping backwards we end up with both players defecting for all rounds.

If the number of rounds is unknown or infinite we have many more Nash equilibria. This is because backward induction no longer works since there is no last round to start at. Then, $s = (D, D)$ is still a Nash equilibrium, but now there are others as well. For example the strategy profile where both players play the tit-for-tat strategy described above.

Nash equilibria got their name from John Nash who made fundamental contributions to game theory. One of his many important theorems, says that every game with a finite number of players and action profiles has at least one Nash equilibrium. [6]

2.3 Cooperative game theory

In the cooperative sub-field of game theory we look at games from another level of abstraction. Players can now form binding coalitions with each other, and each coalition has a payoff associated with it. In this section I make the assumption that payoffs can be freely distributed between coalition members. Players naturally want to know how much of the payoff they are going to get before joining a coalition. In cooperative game theory we mainly study how coalitions are formed and how they divide their payoff, instead of the individual actions we study in non-cooperative game theory. There seems to be some differences in naming cooperative game theory; for example, some call it coalitional game theory instead [3].

2.3.1 Coalitional games

Similarly to how we have the normal-form representation of a game in non-cooperative game theory, in the cooperative sub-field we have coalitional games which can be defined as follows:

The definition of a coalitional game with transferable payoffs is a tuple (N, v), where:

- N is a finite set of n players, indexed by i

- $v : 2^N \mapsto \mathbb{R}$ maps each coalition $S \subseteq N$ to a real valued payoff that can be distributed between the coalition members. We call $v(S)$ the payoff or characteristic function and assume that $v(\emptyset) = 0$.

2.3.2 Voting game

A voting game is a type of coalitional game where there is a subset of winning coalitions $W \subseteq 2^N$. Each winning coalition gets the same payoff P, which I call the available payoff, and the other coalitions get no payoff. More formally $v(S_1) = v(S_2) = P$ for $S_1, S_2 \in W$ and $v(S) = 0$ for $S \notin W$. We also have that only one winning coalition can be formed at a time, or more formally if $S \in W$ then $N \setminus S \notin W$.

I will use political parties in an election as an example. Let there be four political parties A, B, C, and D. The parties have 45, 25, 15, and 15 votes respectively, and the goal is to create a coalition that gets a majority (at least 51) of the votes. The winning coalition will get to decide how to spend a budget of $P = 6$ billion dollars. I chose available payoff $P = 6$ since it is the smallest number divisible by 2 and 3, which is a property that makes for nice numbers in solution concepts later on. Having a small P will be of great interest when the method is introduced, since it reduces the available strategy space which in turn reduces computational time in the evolutionary simulation.

To create a coalition with a majority you need either A plus at least one of the others, or (B, C, D) together. Thus the winning coalitions are (A, B, C, D), (A, B, C), (A, B, D), (A, C, D), (A, B), (A, C), (A, D), and (B, C, D). The structure of the winning coalitions is visualized in figure 2.1.

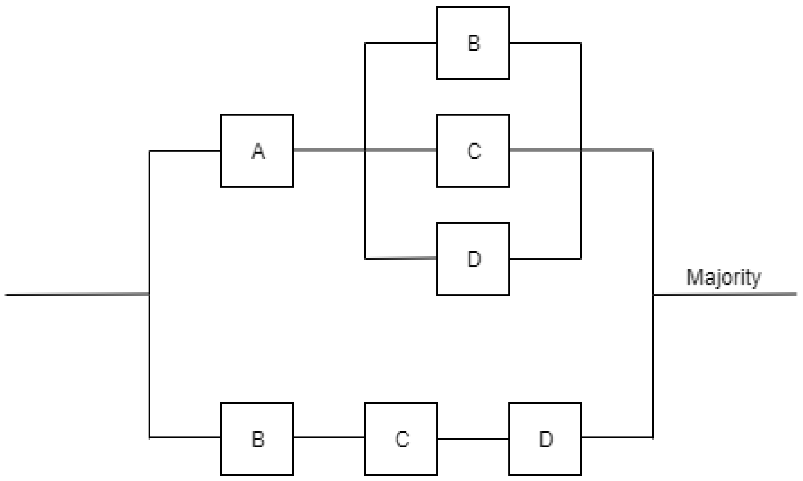


Figure 2.1: Structure of the winning coalitions in the voting game

Notice that even though B brings more votes to a coalition than C or D, it has the same value in the sense you can swap it for either C or D and the resulting coalition will still be winning or still be losing. This notion of a player's value in terms of how it can be swapped with other players will be important when defining solution concepts later.

2.3.3 Analyzing coalitional games

As mentioned previously the topics of study in cooperative game theory are which coalitions will form and how they divide their payoff. When analyzing coalitional games it is common to assume that the grand coalition will form, which means the coalition containing all players. The focus of analysis then becomes how the payoff is divided within the grand coalition. One such payoff division can be called a payoff profile. One of the reasons for the assumption that the grand coalition will form, is that many of the most widely studied games are super-additive games.

A super-additive coalitional game has the property that for each $S, T \subset N$ if $S \cap T = \emptyset$, then $v(S \cup T) \geq v(S) + v(T)$. Which means that the grand coalition will be the coalition that receives the greatest payoff, and thus will likely be formed. This reasoning does not hold for our voting game. The voting game that was defined above is in fact super-additive, but $v(S \cup T) \geq v(S) + v(T)$ can be changed to $v(S \cup T) = v(S) + v(T)$. This means that the grand coalition will get the highest payoff, but all smaller winning coalitions will get the same payoff. Since players are self interested they will most likely prefer the smaller winning coalitions since this would mean less players to share the payoff with.

The solution concepts that will be defined in sections below do assume that the grand coalition will be formed. This might mean that they are less interesting for the analysis of our voting game. Alternatively we might look at a smaller coalition as the grand coalition, but with the players that is not in the coalition receiving a payoff of zero. We will return to this discussion when analyzing results in later chapters.

In the following sections on solution concepts, let $p_i(N, v)$ be the payoff that player i receives by the current payoff profile.

2.3.4 The Shapley value

The Shapley value is the result of trying to find a solution concept that captures fairness. In this context we can define a notion of fairness by defining three axioms a payoff profile should follow in order to be fair. Let us first start by defining the term interchangeable. Two players i, j are said to be interchangable if for all coalitions S that does not contain either i or j, $v(S \cup \{i\}) = v(S \cup \{j\})$. This is the case for players C, B, and D in our voting game. Now for the three axioms:

- **Symmetry**: All interchangeable players should receive the same payoff. More formally, for any v, if i and j are interchangeable then $p_i(N, v) = p_j(N, v)$.

- **Dummy player**: A player i is called a dummy player if i contributes the same amount to any coalition as i could get alone. Which means, $\forall S : i \notin S$, $v(S \cup \{i\}) - v(S) = v(\{i\})$. A Dummy player should receive a payoff of exactly the amount he can get on his own. More formally, for any v, if i is a dummy player then $p_i(N, v) = v(\{i\})$.

- **Additivity**: Consider two different coalitional games (N, v_1), (N, v_2) with two different characteristic functions v_1 and v_2, but with the same set of players N. If we use these games to create a new game where each coalition S gets the payoff $v_1(S) + v_2(S)$, then the players' payoffs in each coalition should be the sum of the payoffs they would have gotten in the separate games. More formally, for any two v_1 and v_2, and for any player i we have $p_i(N, v_1 + v_2) = p_i(N, v_1) + p_i(N, v_2)$.

If we accept these axioms then there exists a unique payoff profile which is called the Shapley value. The Shapley value of player i is defined by:

$$p_i(N, v) = \frac{1}{|N|!} \sum_{S \subseteq N \setminus \{i\}} |S|!(|N| - |S| - 1)!\big[v(S \cup \{i\}) - v(S)\big] \tag{2.2}$$

This expression can seem complicated at first glance. What it is doing is adding the payoff contribution from player i, $v(S \cup \{i\}) - v(S)$, for all different sequences we can create the grand coalition starting from an empty set. The factor $|S|!$ is the number of different ways we can order the set S. The factor $(|N| - |S| - 1)!$ is the number of different ways we can order the remaining players after player i has been added. Dividing by the factor $|N|!$, which is the number of different ways we can order the grand coalition, makes this the average marginal contribution for player i.

Let us compute the Shapley value for our voting game. Since we know that B, C, and D are interchangeable, we can simplify the computations considerably. For example, adding either B, C, or D first has the same result, so we can compute for one of the cases and then multiply that term with 3. The same can be said for (B, C), (B, D), and (C, D). Let us start by computing the Shapley value for player A, who contributes the available 6 payoff to any coalition except (B, C, D) where it contributes 0.

$$
\begin{aligned}
p_A &= \frac{1}{4!}\Big[(3)1!(4-1-1)!(6-0) + (3)2!(4-2-1)!(6-0) + (1)3!(4-3-1)!(6-6)\Big] \\
&= \frac{1}{24}\Big[36 + 36 + 0\Big] \\
&= 3
\end{aligned}
\tag{2.3}
$$

Now for players B, C, and D. Since they are interchangeable we only need to do the computations for one of them. This time I ommit the terms where no contribution is made.

$$
\begin{aligned}
p_B &= \frac{1}{4!}\Big[1!(4-1-1)!(6-0) + 2!(4-2-1)!(6-0)\Big] \\
&= \frac{1}{24}\Big[12 + 12\Big] \\
&= 1
\end{aligned}
\tag{2.4}
$$

Thus the Shapley values are $(3, 1, 1, 1)$, which add up to the available 6 payoff. Let us also compute the Shapley value for a version of the game where the majority requirement is greater, with 80 votes required for a coalition to win. This version of the game will be of special interest when the next solution concept is introduced. The coalitions that achieve at least 80 votes are (A, B, C, D), (A, B, C), and (A, B, D). Observe that both A and B are in all winning coalitions, making them interchangeable in this version, while C and D are still interchangeable. Let us start by computing the Shapley value for player A, who contributes the available 6 payoff to the coalitions (B, C, D), (B, C), and (B, D).

$$
\begin{aligned}
p_A &= \frac{1}{4!}\Big[(2)2!(4-2-1)!(6-0) + (1)3!(4-3-1)!(6-0)\Big] \\
&= \frac{1}{24}\Big[24 + 36\Big] \\
&= 2.5
\end{aligned}
\tag{2.5}
$$

Since the players A and B are interchangeable the Shapley value for player B is the same as for player A. Now let us compute the Shapley value for player C, who contributes available 6 payoff to the coalition (A, B).

$$
\begin{aligned}
p_C &= \frac{1}{4!}\Big[(1)2!(4-2-1)!(6-0)\Big] \\
&= \frac{1}{24}\Big[12\Big] \\
&= 0.5
\end{aligned}
\tag{2.6}
$$

Since the players C and D are interchangeable the Shapley value for player D is the same as for player C. Thus the Shapley values for the greater majority version of the game are $(2.5, 2.5, 0.5, 0.5)$ which of course also add up to the available 6 payoff.

2.3.5 The Core

The core is a solution concept that aims to capture stability, which the Shapley value ignored in favor of fairness. Fairness might have no meaning to self-interested players, and the grand coalition might never be formed. For example in the voting game defined earlier, player A can form a coalition with anyone to create a winning coalition. There is no motivation to include more players than needed to have a winning coalition, since this would mean more players to share the payoff with.

There are games where self-interested players can be incentivized to form the grand coalition, but it is always dependent on the payoff profile. The core is the set of payoff profiles that makes players want to form the grand coalition. A payoff profile is in the core if and only if there is no incentive for any sub-coalitions to break away from the grand coalition. Which means that the sum of payoffs for any sub-coalition $S \subseteq N$ must be at least as big as what they could share among themselves if they broke away from the grand coalition. In other words, a payoff profile p is in the core of a coalitional game (N, v) if and only if

$$\sum_{i \in S} p_i \geq v(S) \quad \text{for} \quad \forall S \subseteq N \tag{2.7}$$

Since the core is a solution concept that aims to capture stability in coalitional games, it is similar to Nash equilibria in non-cooperative game theory. However unlike the guaranteed existence of at least one mixed-strategy Nash equilibria in normal-form games, there is no guarantee that the core is non-empty for coalitional games. In our voting game where a majority of at least 51 votes is needed does have an empty core. As we have discussed earlier, there is no incentive to form the grand coalition since smaller winning coalitions get the same payoff with less players to share it with. If (B, C, D) get less than all of the payoff they have an incentive to break away from the grand coalition, and if A gets zero payoff then he can give the least compensated of (B, C, D) more than he received previously to make him want to join A instead. This back and forth can go on forever if the coalition forming is not limited in some way, more on this later.

If the majority requirement is changed to 80 votes instead, the core is no longer empty. Now the winning coalitions are (A, B, C, D), (A, B, C), and (A, B, D). Any payoff profile that distributes all of the payoff between A and B is now in the core. This is because both A and B are needed to form a winning coalition, and C and D can do nothing on their own. It will be a race to the bottom for the payoffs of C

and D since for any amount of payoff given to one of them, the other would bargain for less in order to be the one chosen for the winning coalition.

2.4 Evolutionary game theory

As mentioned in the introduction, evolutionary game theory is an application of game theory where strategies are treated as individuals in a population and are subjected to evolutionary dynamics. The evolutionary dynamics aim to mimic evolution in nature, where individuals that are well adapted to their environment has a larger chance to spread their genes. We use fitness as a term for the measure of how well an individual is adjusted to environment. The genes of individuals with low fitness will eventually go extinct, and mutations bring new genes for nature to play with. Mutations often result in worse genes, but every now and then a mutation occurs that result in a gene which gives an individual a higher fitness than its surroundings. This individual will then have a higher chance to spread the new gene, and this is how nature creates species that are well adjusted to their environment.

In the context of game theory we can use evolutionary dynamics as a kind of optimization process. Strategies are often initialized randomly, and then we simulate the dynamics that will eventually lead to better strategies. There is a fundamental exploration versus exploitation trade-off in optimization which is also relevant in this context. Exploitation comes from the higher chance of passing on genes for strategies with high fitness. Exploration comes from the mutations that provide new strategies. We can tinker with different parameters in order to influence this trade-off in one way or the other. The focus of evolutionary game theory is often on how prominent strategies change over time and not on only on the resulting strategies when a simulation is finished.

2.4.1 Replicator dynamics

One way to model exploitation is with replicator dynamics. When using replicator dynamics we do not model strategies as individuals, instead we have a population of strategies with associated proportions. The proportion of a strategy is a real number between 0 and 1 which denote how big a part of the population it is, and the sum of all strategies' proportions should always be 1. In order to use replicator dynamics we have to define how to compute fitness for strategies. Let S be the set of all strategies currently in the population indexed by i, and x the vector of corresponding proportions. Replicator dynamics can be used with any normal-form game with a finite amount of players, and let us use a two player game as an example. Let $p_i(j)$ be the payoff strategy i gets when playing versus strategy j. We can then define the fitness $f_i(S)$ for strategy i as follows:

$$f_i(S) = \sum_{j \in S} x_j p_i(j) \tag{2.8}$$

The equation can be modified for games with more players by summing over all combinations of opposing strategies, exchanging x_j for the probability of playing against the current combination, and making p_i a function of more variables. We also need to compute the average fitness:

$$f_{avg}(S) = \sum_{i \in S} x_i f_i(S) \tag{2.9}$$

We can then compute the growth rate for each strategy using the replicator equation:

$$\frac{dx_i}{dt} = r x_i (f_i(S) - f_{avg}(S)) \tag{2.10}$$

Where r is a constant parameter. Strategy proportions are then updated using their growth rates, and strategies with proportions that are too small gets removed.

2.4.2 Mutations

Mutations are used in order to explore the strategy space. There are many ways to do mutations. Usually a strategy is randomly selected with probabilities equal to the strategy proportions. The selected strategy is then used as a starting point for a new, slightly changed strategy that will be added to the population.

3

Method

The aim of the thesis is to study evolutionary dynamics of strategies in the voting game that has been described earlier in the thesis, with the addition of a limited bargaining process. I will give a description of the game and the bargaining process as a reminder. There are four players A, B, C, and D, that have 45, 25, 15, and 15 votes respectively. The objective of the game is to form a coalition that has at least 51 of the votes. There can only be one winning coalition at a time, and the possible winning coalitions are (A, B, C, D), (A, B, C), (A, B, D), (A, C, D), (A, B), (A, C), (A, D), and (B, C, D). The winning coalition gets to divide a payoff of 6 freely between its members. The goal for any individual player is to be a part of the winning coalition and get as much of the available payoff as possible. I have chosen to study this voting game since it is a simple coalitional game with some interesting dynamics described below. It is also quite easy to find real world scenarios where this game is applicable. Maybe most naturally occurring in politics where different parties tries to get a majority in an election.

3.1 Bargaining process

Players naturally want to know how much of the payoff they are going to get before agreeing to join a coalition. This could lead to an infinite bargaining process. If B, C, and D agree to some payoff split, then A can offer the least paid player more than he would get before, and thus convince him to join A instead. Then the other players could offer that player even more to convince him to come back, but then player A can offer someone else more than they would get, and the cycle continues. Thus there is a need to limit the bargaining process. For most real world scenarios that this game could model, there are naturally occurring limiting factors such as a limited amount of time before an election is held. Since different scenarios could lead to model a bargaining process in very different ways I have decided to keep it as simple as possible.

The bargaining process works as follows. Players take turns in proposing a winning coalition and payoff split each, while the proposed players in that coalition gets to vote on if they find their share of the payoff good enough. If all players vote yes, the bargaining process is over and no more proposals are given. If a proposal was agreed upon then that coalition is formed and the players get their shares of the payoff. If no proposal is agreed upon after all players have had a chance to propose, then the available payoff is wasted and all players get nothing. After introducing this bargaining process, the game can be analyzed using both cooperative and non-cooperative game theory. Since the bargaining process limits each players actions the game can be represented as a normal-form game. We can still keep the coalitional game representation and forget about the bargaining process when analyzing the game from a higher level of abstraction.

3.2 Evolutionary simulation

The game will be played in the context of an evolutionary process which I will describe below. Some of the interesting questions are what kind of strategies will be formed, what coalitions will be accepted, and if the population converges to any long term stable strategies.

3.2.1 Strategies

There are two different aspects of the game that needs to be kept in mind when modelling strategies: what to propose and what to vote on other players' proposals. These aspects could be modelled with more or less complexity. I decided on a pretty simple approach to keep the computational complexity lower as well as making it easier to analyze the results.

I chose to model the proposal part as a pair of two proposals, each with a corresponding probability. This is a simple way to allow for mixed strategies, albeit limited mixed strategies. The probabilities must be made sure to always sum to one, even when strategies are mutated. I limited the granularity of probabilities to a single decimal point to once again limit the size of strategy space, this means that 0.1 is the smallest non-zero probability associated with a proposal. Each proposal contains a mapping from players to integer payoffs, where each player in the mapping is a part of the proposed coalition. In my implementation proposals must be of winning coalitions. I made this choice early to make the implementation easier and for the simulation to result in decent strategies quicker, since proposing a non-winning coalition obviously is a sub-optimal strategy.

I chose to model the voting part as a single integer I call minimum payoff, which is the least amount of payoff a player will vote yes on. Any amount that is equal to or higher than the minimum payoff will always get a yes vote, and any lower amount a no vote. Below is an example of a strategy with minimum payoff 2, and two different proposals with probabilities 0.3 and 0.7 respectively. The proposal with 0.3 probability of being selected is $\{A : 4, B : 2\}$, which proposes to form the winning coalition (A, B), with player A given 4 payoff, and player B given 2. The proposal with 0.7 probability of being selected is $\{B : 2, C : 2, D : 2\}$, which proposes to form the winning coalition (B, C, D), with each player given an equal payoff of 2.

$2, (\{A : 4, B : 2\}, \{B : 2, C : 2, D : 2\}), (0.3, 0.7)$

I chose to work with only integers, both for proposals and minimum payoffs. The biggest reasons are that it is easier to implement and it reduces the size of the strategy space, but it is a limitation of the model. Another limitation is that I do not take any proposal history into account. Both minimum payoffs and proposals could be functions that take the history as an input, to allow for more complex strategies. This change could be very interesting, especially when combined with a more complex bargaining process.

Another choice I made was to separate strategy populations into different populations for each position A, B, C, and D. The strategy example above seems like a reasonable strategy for a strategy playing as B. Both proposals gives B non-zero payoff, and the proposed coalitions does not contain any unnecessary players for them to be winning coalitions.

I did not start with separate strategy populations. Initially all strategies played for all positions. The first version of a strategy model was a minimum payoff and a single proposal. I quickly realized that this was inadequate when strategies played for all positions, since a good strategy for playing as A is not a good strategy when playing as B, C, or D. This lead me to make the strategy model more complicated. Each strategy got a minimum payoff for each position, as well as a proposal for each position. This meant that more reasonable strategies could be formed with different proposals and minimum payoffs for different positions, but the results were hard to analyze, since the evolutionary dynamics could make a strategy dominant in the population even if the strategy contained irrational proposals or minimum payoffs for one of the positions. One of the irrational types of proposals were proposals that contained coalitions without the player itself. This could happen since the fitness of a strategy was averaged over all positions.

This led me to separating the strategy populations, since now strategies gets much more punished when playing irrationally, and can not rely on being good for other positions. The first version of separated strategies did not allow for mixed strategies. I decided to give strategies the option of being mixed, since mixed strategies are prominent in most games. As mentioned in earlier every game is guaranteed to have a mixed strategy Nash equilibria, but not guaranteed to have any pure strategy equilibria. Strategies can still be pure, which is the case when one of the proposal probabilities is 1, or when the proposals are identical.

3.2.2 Size of strategy space

Knowing the size of the strategy space is of interest when evaluating whether some functions are computationally feasible, as well as when analyzing results. This subsection will be devoted to computing this size. Let us denote the strategy space by S where $|S|$ is the size of this space. Let us also denote the set of all minimum payoffs by M, the set of all proposals by P, and the set of all proposal probabilities by Q. Then we can compute $|S|$ as follows:

$$|S| = |M||P|^2|Q| \tag{3.1}$$

To make the computation simpler I divide the problem into smaller parts. Since minimum payoffs are integers from 0 to 6 we have:

$$|M| = 7 \tag{3.2}$$

The proposal probabilities must sum to 1, thus one of the proposal probabilities can be inferred from the other. This fact combined with the knowledge that proposal probabilities only increase or decrease in increments of 0.1 gives us:

$$|Q| = 11 \tag{3.3}$$

The number of proposals is trickier to compute. Let us denote all proposals involving two players by P_2, three players by P_3, and four players by P_4. We can split the computation into three parts to make it simpler:

$$|P| = |P_2| + |P_3| + |P_4| \tag{3.4}$$

Now let us denote the set of all winning coalitions involving two players by C_2, three players by C_3, and four players by C_4. Let us also denote the set of all ways to divide the available payoff for coalitions involving two players by D_2, three players by D_3, and four players by D_4. Since only proposals of winning coalitions are allowed we have:

$$\begin{aligned}
|P_2| &= |C_2||D_2| = 3 \times 7 = 21 \\
|P_3| &= |C_3||D_3| = 4 \times 28 = 112 \\
|P_4| &= |C_4||D_4| = 1 \times 84 = 84
\end{aligned} \tag{3.5}$$

Thus:

$$|P| = |P_2| + |P_3| + |P_4| = 21 + 112 + 84 = 217 \tag{3.6}$$

Bringing it all together:

$$|S| = |M||P|^2|Q| = 7 \times 217^2 \times 11 = 3625853 \tag{3.7}$$

3.2.3 Initialization

The simulation has a number of different parameters that are initialized before it starts:

- **num_iterations**: The number of iterations the simulation is run before terminating. A normal value is 100000.
- **max_num_strategies**: The maximum number of strategies that each position's strategy population can contain. This parameter is used to lower computational complexity. A normal value is 5.
- **num_starting_strategies**: The number of random strategies that are initialized for each position's strategy population. This parameter must be lower than or equal to max_num_strategies. A normal value is 5.
- **r**: This is the constant parameter used in the replicator equation (equation 2.10). A normal value is 0.2.
- **min_proportion**: The lowest proportion of a strategy before it is removed. This parameter set to a small value greater than zero. Removing strategies when their proportions are less than or equal to zero can cause removal of low fitness strategies to be very slow. This is because the replicator equation (equation 2.10) contains a proportion factor. A normal value is 0.005.
- **num_mutations_per_iteration**: The number of mutations that occur in each strategy population every iteration. A normal value is 1.
- **new_mutation_proportion**: How much proportion should mutated strategies start with. A normal value is 0.01.
- **available_payoff**: It is the payoff a winning coalition gets to divide between its members. As in the description of the voting game, I have used 6.
- **voting_powers**: The number of votes the players (A, B, C, D) bring to a coalition. As in the description of the voting game, I have used $(45, 25, 15, 15)$.

- **voting_power_needed**: The number of votes (voting_power) needed for a coalition to be winning. I have used 51 and 80.

After the parameters are loaded the four strategy populations are initialized with random strategies. Random strategies have random minimum payoffs, random proposals, and random proposal probabilities. A random minimum payoff is simply a random integer between 0 and 6. A random proposal is created by first selecting one of the winning coalitions at random, then randomizing a valid payoff split between the players in that coalition. The first random proposal probability is created by generating a random integer between 0 and 10 and dividing that number by 10. The second proposal probability is simply the result of subtracting the number 1 with the first probability.

3.2.4 Computing fitness

After the initialization steps the simulation starts the iterating process. What happens first in each iteration is the computation of fitness values for each strategy, as well as computing the average fitness. The fitness values are computed by iterating over every combination of strategies from the different populations. For every such combination every combination of proposal indices iterated through, and one game is played with the current strategy combination and proposal indices combination. By proposal indices I mean a vector containing ones and zeros that denote which of the two proposals will be selected by each position.

One game being played means that strategies go through the bargaining process for each proposal order, and payoffs are accumulated and averaged over the number of such orders. There could be that no coalition gets accepted. In that case no payoffs are accumulated. Resulting payoffs from a game is not added to the fitness immediately. Instead the payoff of a strategy is multiplied by the probability of facing the current opposing strategies. This probability is the multiplied proportions of the opponents. Since all combinations of proposal indices are played, we have to also multiply with the probability of seeing the current proposal index combination. This probability is the multiplied proposal probabilities associated with the proposal indices.

Let S_B, S_C, and S_D be the sets of strategies currently in populations B, C, and D. Let C be the set of all combinations of proposal indices and $P(c)$ the probability of combination c. Let x_j, x_k, and x_l be the proportions of strategies j, k, l, and $p_{i,A}(j, k, l)$ the payoff received by strategy i when playing position A versus strategies j, k, l in positions B, C, D. Then the fitness for a strategy i in population A can be computed by:

$$f_{i,A} = \sum_{j \in S_B} \sum_{k \in S_C} \sum_{l \in S_D} \sum_{c \in C} x_j x_k x_l P(c) p_{i,A}(j, k, l) \tag{3.8}$$

I realize that this description might be hard to follow, and I will provide a simplified pseudo code implementation of the fitness computing function to hopefully make it clearer. The strategies given as data to the algorithm is a vector of four vectors, one for each position. This means that $strategies[0]$ is the strategy population vector for position A, $strategies[1]$ is the strategy population vector for position B, and so on. Similarly, the proportions given as data is a vector of vectors with the same dimensions as strategies. Every element in proportions is the proportion for the corresponding strategy in strategies.

Algorithm 1: Compute Fitness

```
Data: strategies, proportions
Result: Fitness vector of vectors with the same dimensions as strategies, and
        average fitness vector of dimension 4
Initialize fitness and average fitness zeros for all elements
Initialize proposal_index_permutations to all binary strings of length 4
for strategy i in strategies[0] (population A) do
    for strategy j in strategies[1] (population B) do
        for strategy k in strategies[2] (population C) do
            for strategy l in strategies[3] (population D) do
                players = (strategies[0][i], strategies[1][j], strategies[2][k],
                  strategies[3][l])
                for proposal_indices in proposal_index_permutations do
                    compute proposal_indices_probability
                    if proposal_indices_probability $\neq$ 0 then
                        payoffs = play_game(players)

                        proportion_product = proportions[0][i] * proportions[1][j] *
                          proportions[2][k] * proportions[3][l]

                        fitness[0][i] += payoffs[0] * (proportion_product /
                          proportions[0][i]) * proposal_indices_probability
                        fitness[1][j] += payoffs[1] * (proportion_product /
                          proportions[1][j]) * proposal_indices_probability
                        fitness[2][k] += payoffs[2] * (proportion_product /
                          proportions[2][k]) * proposal_indices_probability
                        fitness[3][l] += payoffs[3] * (proportion_product /
                          proportions[3][l]) * proposal_indices_probability
                    end
                end
            end
        end
    end
end
for positions p do
    average_fitness[p] = dot_product(fitness[p], proportions[p])
end
```

An important implementation detail is that results from games are stored, which leads to greatly reducing computational complexity. This is one of the implementation details that is not shown in the pseudo code above.

3.2.5 Replicator dynamics

After the fitness is computed for all strategies, it is used for the replicator dynamics. The replicator dynamics affect the strategy populations separately, but in the same way. The proportion growth rates are computed for each strategy according to the replicator equation (equation 2.10). The proportions are then updated by addition with their respective growth rates. After updating the proportions all strategies that have less proportion than the parameter *min_proportion* are removed. The proportions can sum to a number different from one, thus the last step of the replicator dynamics is to normalize the proportions by simply dividing element wise with the sum of the proportions.

3.2.6 Mutations

After the replicator dynamics has affected strategy populations, it is time for mutations to give populations new strategies to work with. Mutations are done separately for the different populations, just like the replicator dynamics. Each population does a number *num_mutations_per_iteration* of mutations each iteration, which is normally set to one. There are two cases before a mutation is performed. Either the strategy population is at its limit, meaning that the number of strategies is equal to the parameter *max_num_strategies*, or there is space for new strategies.

If the number of strategies is less than *max_num_strategies*, then a strategy is randomly selected, weighted by proportion. In other words, if there are two strategies in a population, one with 0.8 proportion and the other with 0.2 proportion, then the first strategy would be selected 80% of the time, and the other 20% of the time. The selected strategy will then be mutated and the new strategy will be given a proportion equal to the parameter new_mutation_proportion, which is subtracted from the selected strategy's proportion.

If the number of strategies is equal to *max_num_strategies*, then the strategy with the least amount of proportion will be automatically selected. The selected strategy will then be mutated, and the result of the mutation will replace the selected strategy, completely taking over its proportion.

Now I will explain how a mutation works. First a property of the selected strategy is randomly selected. This is done by uniform random selection between mutating the minimum payoff, the proposals, or the proposal probabilities.

If the minimum payoff is selected there are a few cases to consider. If the minimum payoff is zero then it is automatically changed to one. If it is equal to the available payoff it is automatically changed to the available payoff minus one. If none of these conditions hold, the minimum payoff gets changed by either adding or subtracting one, which is selected randomly with equal probability.

If the proposals are selected then one of the proposals is randomly selected with equal probability. The selected proposal has its payoff split mutated, or is replaced by a completely new random proposal with equal probability. If the payoff split is selected then two players of the proposed coalition are randomly selected and one payoff is given from one player to another. If one of the players gets all of the available payoff, then that player automatically is the one giving. Otherwise the giving player is randomly selected with equal probability.

If the proposal probabilities are selected then 0.1 probability is taken from one of the proposals and given to the other. Similarly to mutating the payoff split of a proposal, if one of the probabilities is one, then that probability automatically gives to the other. Otherwise the giving player is randomly selected with equal probability.

3.2.7 Note on randomness

The model underwent many iterations, and one of aspects that changed the most is how fitness is calculated. I started out with sampling, where a number of games were played with randomly selected strategies. This made fitness vary significantly between calculations, even with relatively large sample sizes. Strategies were randomly selected weighted by their proportions, thus the fitness for strategies with small proportions varied the most.

Having fitness values vary from randomness leads to replicator dynamics affecting proportions in the wrong way. This leads to problems when analyzing population statistics over time, since it is hard to know if oscillations come from evolutionary dynamics or randomness in fitness values. This lead to computing fitness by playing all combinations of strategies instead.

Even after playing all combination of strategies there were still some randomness in fitness calculations. One of the issues were that proposal orders were randomized instead of playing all of them and averaging. Another issue was that when mixed strategies first were implemented, the proposals were selected randomly, weighted by their proposal probability. I eventually removed all randomness from fitness calculations, which lead to larger computational complexity. This made it important to optimize the code in different ways, such as storing game results. In the end it provided a big benefit, since one can now be certain that oscillations in strategy populations comes from randomness in the evolutionary dynamics and not from varying fitness values.

4

Results

In this chapter I am going to show results from the evolutionary simulation, and compare those results with some of the theory. I have run simulations with many different parameter settings, but I will focus on simulations with two different sets of parameters. The sets share every parameter setting except *voting_power_needed*, which will take on the values 51 and 80. I choose to try different values for this parameter as it affects the theoretical solution concepts in an interesting way. The parameter settings that I have used are:

- **num_iterations**: 100000.
- **max_num_strategies**: 5.
- **num_starting_strategies**: 5.
- **r**: 0.2.
- **min_proportion**: 0.005.
- **num_mutations_per_iteration**: 1.
- **new_mutation_proportion**: 0.01.
- **available_payoff**: 6.
- **voting_powers**: $(45, 25, 15, 15)$.
- **voting_power_needed**: 51 or 80, which will be specified.

I chose these parameter values based on trial and error, where I looked for reasonable running time, and how long it took to converge to relatively stable populations.

In almost all of the simulations the populations eventually converged to pure strategies. A reasonable explanation for this phenomenon is that mixed, non-pure strategies are mostly used in games in order to avoid getting exploited by opponents.

Because of its cooperative structure there is no real way of exploiting others in the voting game, instead there is incentive to find cooperation with a sub-set of the other players. Cooperation is established in the form of aligning ones minimum payoff with others' proposals, and vice versa.

There are different ways to visualize the results of a simulation. One of the ways is to plot the percentage of wins each feasible coalition gets over time. Feasible coalitions are the set of coalitions that can result from playing a game. This means that it is the set including all coalitions that result in a majority with the empty coalition also included. I will use these plots to visualize simulation results often, since it is a good way to characterize the prominent strategies of the populations. A limitation of this type of plot is that it captures what proposals are accepted, but it does not show the minimum payoffs or payoff splits by its own. To get more details I will combine the plots with text data that describes the strategies fully.

4.1 Simple majority ($voting\_power\_needed = 51$)

In this section I show results from simulations where a winning coalition needs to have at least 51% of the votes (which is the same as having 51 voting power).

The results can be divided into two different categories, one where a stable state of winning coalitions are found and lasts for the rest of the simulation, and another where this is not the case. In the latter category, periods of stability are observed, but there are multiple different stable periods in the same simulation. Between these periods there are big changes that usually happens quickly. In all simulations, after a small amount of iterations, a game never results in the empty coalition. In other words all games result in some proposal getting all yes votes after a short amount of evolutionary iterations. A sort of cooperation is observed, were minimum payoffs and proposals for different positions align in a way such that most proposals get accepted immediately.

4.1.1 Stable states

There are different types of stable states the simulation can end up in. I am going to show two of them in this subsection. The divisions of coalition wins that are shown in the examples below are not sufficient requirements for stability, but when stability is observed these are two of a low number of possibilities.

First example

The first example is a state where one coalition wins 1/2 of the games and two other coalitions both win 1/4 of the games. A plot of a simulation where such a stable state is established can be seen in figure 4.1, where coalition (A, C) wins 1/2 of the time, (A, B) wins 1/4 of the time and (B, C, D) wins the remaining 1/4 of the time.

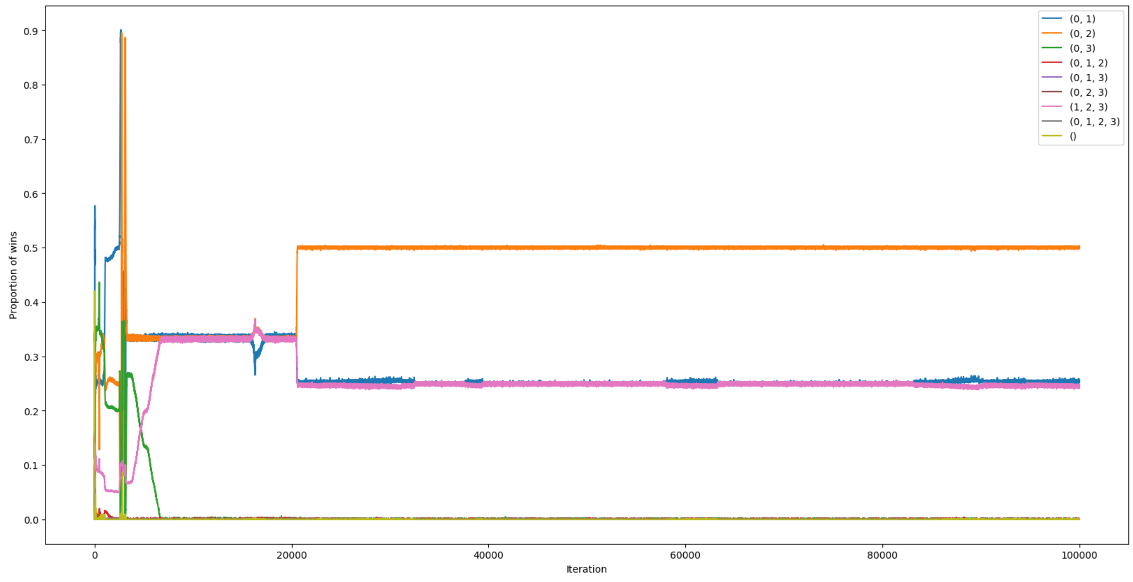


Figure 4.1: Plot of proportion of wins for each coalition over time, for a simulation that ends up in a stable state.

All of the coalitions without unnecessary players, which is (A, B), (A, C), (A, D), and (B, C, D), can be a part of the trio of stable winning coalitions. It is typical that the prominent strategies in position A proposes a coalition that include itself and one of the other positions, and the prominent strategies for that position reciprocate by proposing the same coalition. The payoff splits and minimum payoffs are such that both proposals gets immediately accepted, which leads to that coalition winning 1/2 of the games, since player A or the other player are the first to propose in 1/2 of the games.

The other positions has strategies that propose coalitions including itself and either position A or the other positions in coalition (B, C, D). The payoff splits and minimum payoffs are such that the proposals immediately gets accepted, leading to the respective coalitions winning 1/4 of the time each.

In figure 4.1 we can see that (A, C) wins 1/2 of the time, thus we can deduce that both positions A and C has strategies that propose this coalition. We can also see that (A, B) and (B, C, D) wins 1/4 of the games each, which means that position B has strategies that propose (A, B), and position D has strategies that propose (B, C, D).

To give a more detailed view into the strategies for each position, I will display the resulting strategies after the final iteration of the simulation that produced the plot in figure 4.1:

Table 4.1: Position A

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:1, B:3, C:2	A:5, C:1	0.0	1.0	0.48285	3.27484
4	A:1, B:2, C:3	A:5, C:1	0.0	1.0	0.42234	3.27484
4	A:4, B:2	A:5, C:1	0.0	1.0	0.07292	3.27484
2	A:1, B:3, C:2	A:5, C:1	0.0	1.0	0.01381	3.27484
4	A:0, B:3, C:3	A:5, C:1	0.0	1.0	0.00808	3.27484

Table 4.2: Position B

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
2	A:0, B:2, D:4	A:4, B:2	0.0	1.0	0.53885	0.99792
2	A:0, B:1, D:5	A:4, B:2	0.0	1.0	0.40760	0.99792
2	A:1, B:2, D:3	A:4, B:2	0.0	1.0	0.03651	0.99792
2	A:6, B:0	A:4, B:2	0.0	1.0	0.01176	0.99792
3	A:0, B:1, D:5	A:4, B:2	0.0	1.0	0.00527	0.67224

Table 4.3: Position C

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:0, B:2, C:2, D:2	A:4, C:2	0.0	1.0	0.85648	0.99999
1	A:3, B:1, C:2, D:0	A:4, C:2	0.0	1.0	0.07561	0.99999
1	A:4, D:3	A:4, C:2	0.0	1.0	0.04255	0.99999
3	A:3, B:3, C:0, D:0	A:4, C:2	0.0	1.0	0.01671	0.99999
1	B:1, C:3, D:2	A:4, C:2	0.0	1.0	0.00865	0.99999

Table 4.4: Position D

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
2	B:2, C:1, D:3	A:5, C:1	1.0	0.0	0.69576	0.73357
2	B:2, C:1, D:3	A:0, B:2, C:4	1.0	0.0	0.19002	0.73357
2	B:2, C:1, D:3	A:5, B:0, C:1	1.0	0.0	0.05617	0.73357
2	B:2, C:1, D:3	A:5, B:0, C:1	1.0	0.0	0.05170	0.73357
3	B:3, C:0, D:3	A:0, B:2, C:4	1.0	0.0	0.00635	0.00000

Notice that each position has evolved to playing essentially one strategy. There are tiny deviations that does not affect fitness, except for a few outliers that affect fitness very little. Every position has only pure strategies since the proposal probabilities are either zero or one. This means that only one of the proposals are relevant when it comes to computing fitness. Thus most strategies for a given position are essentially the same strategy. For example look at table 4.1, which displays strategies for position A. We can see that the second proposal is the only one that is proposed, and each strategy has the same second proposal. Thus the differences in the first proposal does not matter for the current fitness calculations, and the only currently relevant difference between strategies is that the fourth strategy has a different minimum payoff than the others. This strategy still has the same fitness as the others, since there are no relevant proposals from other positions that allocates less than 4 payoff to A.

We can see that a sort of cooperation between positions is established. Notice that the relevant proposal from position A, $(A : 5, C : 1)$, is aligned with the minimum payoffs for position C which is 1 for all strategies except for one strategy with very low proportion. The relevant proposal from position C, $(A : 4, C : 2)$, is aligned with the minimum payoffs for position A. Similar alignment between proposals and minimum payoffs can be seen for the other positions. Position B proposes $(A : 4, B : 2)$ which gives A exactly the minimum payoffs most of position A demands. Position D proposes $(B : 2, C : 1, D : 3)$ which aligns with the minimum payoffs for both positions B and C.

Second example

Another type of stable state is a state where the four coalitions that does not contain unnecessary players ((A, B), (A, C), (A, D), and (B, C, D)), win the game equally often, and no other coalition ever wins. In other words, 1/4 of the games result in one of the four coalitions, 1/4 of the games in another, and so on. An example of such a state can be seen in figure 4.2.

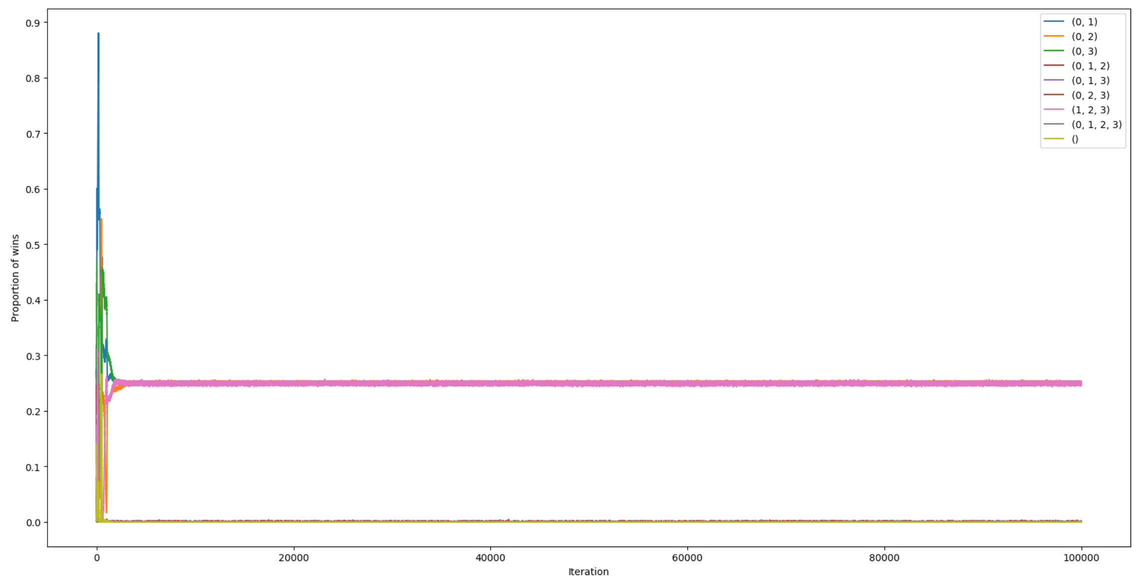


Figure 4.2: Plot of proportion of wins for each coalition over time, for a simulation that ends up in a stable state.

I will display the resulting strategies after the final iteration of the simulation that produced the plot in figure 4.2:

Table 4.5: Position A

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:5, C:1	A:5, C:1	0.8	0.2	0.79309	3.25835
4	A:5, C:1	A:5, C:1	0.6	0.4	0.11239	3.25835
4	A:5, C:1	A:5, C:1	0.7	0.3	0.05697	3.25835
3	A:5, C:1	A:5, C:1	0.6	0.4	0.03130	3.25835
4	A:6, C:0	A:5, C:1	0.8	0.2	0.00625	2.79296

Table 4.6: Position B

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:3, B:3	A:4, B:2	0.0	1.0	0.80358	0.75060
1	A:2, B:4	A:4, B:2	0.0	1.0	0.08465	0.75060
0	A:3, B:3	A:4, B:2	0.0	1.0	0.07517	0.75060
1	A:5, D:1	A:4, B:2	0.0	1.0	0.02773	0.75060
0	A:3, B:3	A:4, B:2	0.1	0.9	0.00888	0.71079

Table 4.7: Position C

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:2, B:3, C:1	B:1, C:2, D:3	0.0	1.0	0.79285	0.74651
1	A:2, B:4, C:0	B:1, C:2, D:3	0.0	1.0	0.16416	0.74651
1	A:2, C:4	B:1, C:2, D:3	0.0	1.0	0.02273	0.74651
1	A:5, B:1, C:0, D:0	B:1, C:2, D:3	0.0	1.0	0.01358	0.74651
1	A:3, B:2, C:1	B:1, C:2, D:3	0.0	1.0	0.00668	0.74651

Table 4.8: Position D

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
3	A:4, D:2	A:3, C:3	1.0	0.0	0.75801	1.25246
2	A:4, D:2	A:3, C:3	1.0	0.0	0.10631	1.25246
3	A:4, D:2	A:2, C:4	1.0	0.0	0.08956	1.25246
3	A:4, D:2	A:3, C:1, D:2	1.0	0.0	0.03824	1.25246
4	A:4, D:2	A:3, C:3	1.0	0.0	0.00788	0.66863

Again we can see that the strategy populations has evolved to essentially a single pure strategy per position. Even though the proposal probabilities for position A are not ones and zeros, the proposals are the same so the probabilities does not matter. There is a strategy for each position that differs from the rest in fitness, but those strategies all have less fitness than the others and are about to get replaced.

Notice that again the proposals and minimum payoffs are aligned between positions, such that every proposal immediately gets accepted. Since every position proposes a different coalition, we can see that this leads to 1/4 of games resulting in those coalitions winning, as seen in figure 4.2.

4.1.2 Unstable states

Some of the simulations does not settle into stable states that holds for the remainder of the iterations. An example of such a simulation produced the plot in figure 4.3.

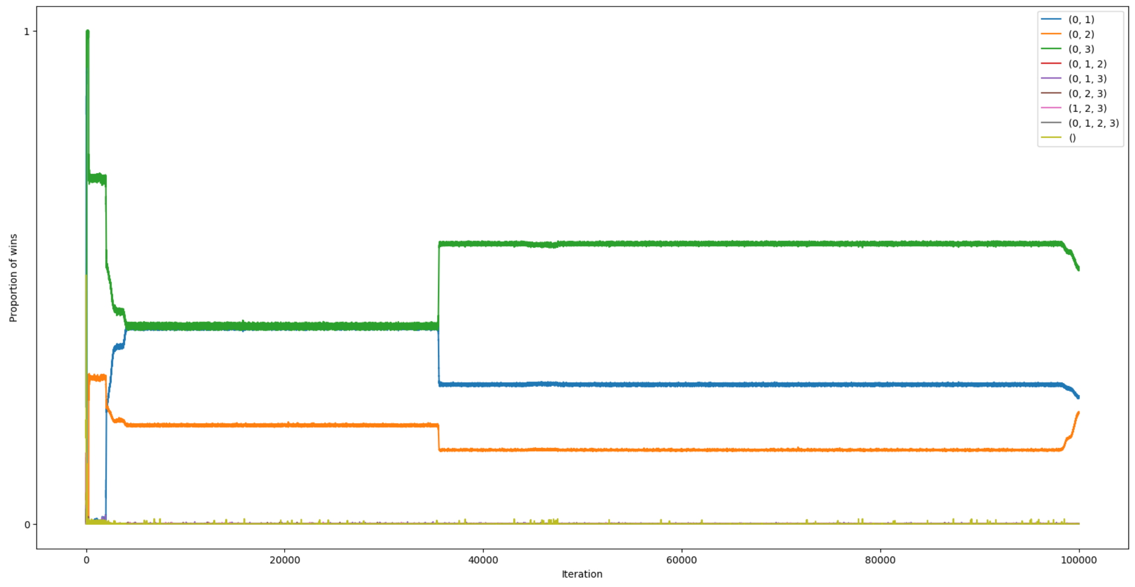


Figure 4.3: Plot of proportion of wins for each coalition over time, for a simulation that does not end up in a stable state.

Observe that there are long periods of stability, but that in the end there is a big change. The coalitions are moving towards 1/2, 1/4, and 1/4 of wins each similarly to the first example of a stable state, and might have stayed there if the simulation continued. I will show the final strategies of the simulation:

Table 4.9: Position A

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:0, D:6	A:4, D:2	0.0	1.0	0.81773	3.99992
4	A:3, B:2, C:1	A:4, D:2	0.0	1.0	0.11625	3.99992
4	B:4, C:0, D:2	A:4, D:2	0.0	1.0	0.03174	3.99992
3	A:0, D:6	A:4, D:2	0.0	1.0	0.02689	3.99986
5	A:0, D:6	A:4, D:2	0.0	1.0	0.00739	3.99992

Table 4.10: Position B

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:0, C:6	A:4, B:2	0.0	1.0	0.65520	0.51220
1	B:5, C:1, D:0	A:4, B:2	0.0	1.0	0.25641	0.51221
5	A:0, C:6	A:4, B:2	0.0	1.0	0.05098	0.51220
2	A:0, C:6	A:4, B:2	0.0	1.0	0.03196	0.51221
4	A:0, C:6	A:4, B:2	0.1	0.9	0.00545	0.51220

Table 4.11: Position C

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:4, C:2	A:5, D:1	0.9	0.1	0.86005	0.44679
4	A:4, C:2	A:5, D:1	1.0	0.0	0.08312	0.49643
4	A:4, C:2	A:5, D:1	0.8	0.2	0.03578	0.39714
5	A:4, C:2	A:5, D:1	0.9	0.1	0.01270	0.44679
3	A:4, C:2	A:5, D:1	0.9	0.1	0.00834	0.44679

Table 4.12: Position D

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
2	A:4, D:2	A:3, B:2, D:1	1.0	0.0	0.70879	1.03890
2	A:4, D:2	A:3, B:3	1.0	0.0	0.25627	1.03890
2	A:4, D:2	A:2, B:4, C:0	1.0	0.0	0.01655	1.03890
2	A:4, D:2	A:1, D:5	1.0	0.0	0.01044	1.03890
2	A:4, D:2	A:3, B:3	0.9	0.1	0.00794	1.00567

Notice that fitness values are largely the same within positions as seen in the stable examples, this time however there are significant differences for position C. This difference in fitness leads to changes in proportions, which in turn explains the instability in coalition wins we observe in figure 4.3.

The fact that the simulation had long periods of stability that eventually changed, opens up the question of the actual long term stability of the examples of stable states in the previous subsection. This is one of the questions that will be discussed in the next subsection.

4.1.3 Comparison with theory

Cooperative game theory

I will start by comparing the results with solution concepts from cooperative game theory. The core is empty so of course we cannot observe any results relevant to this concept. The Shapley value is not observed as a proposal in any results. This is natural since the Shapley value aims to capture fairness, which is a concept that the the fitness maximizing evolutionary dynamics does not explicitly care about. As mentioned earlier, strategies have no incentive to involve more players than necessary in proposals, which leads to the grand coalition never forming.

However, when the final fitness values are averaged over simulations an interesting result is observed. I accumulated the final fitness values of 30 simulations for the top strategy of each position and normalized. The result was

$$(3.85218, 0.70165, 0.71548, 0.73550) \tag{4.1}$$

for the positions (A, B, C, D), which we can compare to the Shapley value $(3, 1, 1, 1)$. It is natural that we observe greater fitness values for position A, since it is a part of more winning coalitions. It is also natural that fitness values for positions B, C, and D are approximately the same, since the positions are interchangeable when it comes to forming winning coalitions. It is reasonable to assume that the values for B, C, and D would get even closer to each other if we averaged over a greater number of simulations. The reason for not running more simulations is that they take a long time to finish.

Non-cooperative game theory

I have raised the question about the actual long term stability of the examples of seemingly stable states. To answer this question non-cooperative game theory is of greater relevance, more specifically the solution concept Nash equilibrium. If the prevalent strategies for every population is forming a Nash equilibrium then none of these strategies has any incentive to change, since it means that they can not get a higher fitness than they currently have. In other words, no mutation can occur that has a higher fitness, which means that no mutation can take over and become the prevalent strategy in a population.

Note that populations could be stable for a long time even if the prevalent strategies does not form a Nash equilibrium. There might be very few better strategies, or mutations might not be able to create the better strategies, since it might take many mutations on the way to the better strategy and the mutations might not survive on the way. Considering how mutations work and the size of the strategy space, the vast majority of possible strategies will never be tried.

To check if the populations are in a Nash equilibrium, I will test all possible strategies for every position, one at a time, while the strategies for the other positions remains the same. If a strategy is found that gets a higher fitness than any of the current strategies, then we can conclude that the prevalent strategies does not form a Nash equilibrium. If no such strategy is found we can conclude that they do form a Nash equilibrium and predict that populations will be stable in the long run.

Let us start with the first example of a stable state, where the final strategies resulted from the simulation that produced the plot in figure 4.1. I found that not all positions played strategies that were best response strategies. Position B should switch to a strategy that proposes $(B : 3, C : 1, D : 2)$ as a pure strategy instead of the current proposal $(A : 4, B : 2)$. The new proposal will not always be accepted since some of the strategies for positions C and D have minimum payoffs that are too high, but those strategies have a very small proportion, and the new proposal will be accepted enough of the time to make the better payoff split worth it. If position B changed to this proposal its fitness would be 1.22131 which is better than the fitness of the current prevalent strategies 0.99792. Thus the prevalent strategies are not forming a Nash equilibrium, and the populations would lose their stability as soon as position B got a mutation with the proposal $(B : 3, C : 1, D : 2)$ with a sufficiently high corresponding proposal probability.

Now let us analyze the second example of a stable state, where the final strategies resulted from the simulation that produced the plot in figure 4.2. Once again there is no Nash equilibrium. Position D should switch to a strategy that proposes $(B : 1, C : 1, D : 4)$ as a pure strategy. This proposal will always get accepted since the minimum payoffs are either 0 or 1 for the strategies in positions B and C. Changing to this proposal would result in a fitness value of 1.75344, instead of the old value 1.25246.

4.2 Greater majority ($voting\_power\_needed = 80$)

In this section I show results from simulations where a winning coalition needs to have at least 80% of the votes (which is the same as having 80 voting power).

For this version of the game there are only three coalitions that achieve a sufficient majority. Those coalitions are (A, B, C), (A, B, D), and (A, B, C, D). As one might expect the coalition (A, B, C, D) is never observed in the results, just like the coalitions with unnecessary players in the simple majority game version. The empty coalition also does not win a significant amount of games after a few evolutionary iterations. This leaves the coalitions (A, B, C), and (A, B, D) which all results involve.

The majority of simulations end up with 1/2 of all games being won by (A, B, C) and the other 1/2 by (A, B, D). A plot of such a simulation can be seen in figure 4.4.

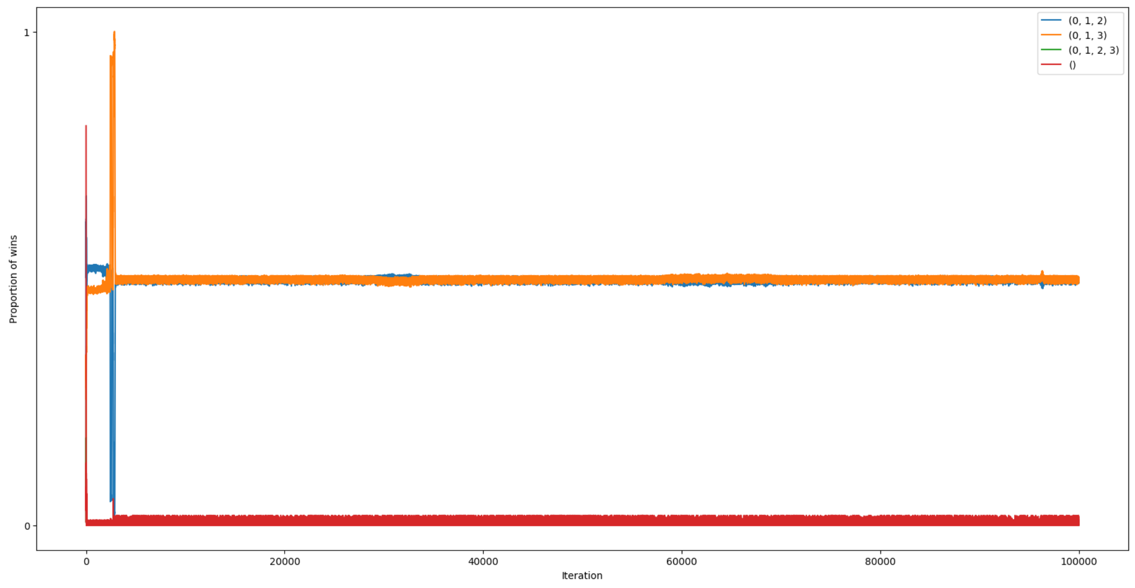


Figure 4.4: Plot of proportion of wins for each coalition over time, for a simulation with a greater majority requirement.

I will show the final strategies for the simulation that produced the plot in figure 4.4:

Table 4.13: Position A

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
4	A:5, B:1, C:0, D:0	A:5, B:1, C:0	0.3	0.7	0.90692	3.99996
4	A:4, B:1, C:0, D:1	A:5, B:1, C:0	0.3	0.7	0.04275	3.99996
4	A:5, B:1, C:0, D:0	A:5, B:1, C:0	0.4	0.6	0.02823	3.99996
4	A:5, B:1, C:0, D:0	A:5, B:1, C:0	0.2	0.8	0.01210	3.99996
5	A:5, B:1, C:0, D:0	A:5, B:1, C:0	0.2	0.8	0.01000	0.00000

Table 4.14: Position B

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:3, B:3, C:0, D:0	A:3, B:2, D:1	0.3	0.7	0.53134	0.98999
1	A:5, B:1, C:0	A:4, B:2, D:0	0.2	0.8	0.28182	0.98999
1	A:3, B:3, C:0, D:0	A:4, B:2, D:0	0.3	0.7	0.09901	0.98999
1	A:3, B:3, C:0, D:0	A:4, B:2, D:0	0.2	0.8	0.08140	0.98999
1	A:5, B:1, C:0, D:0	A:3, B:2, D:1	0.3	0.7	0.00643	0.98999

Table 4.15: Position C

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:4, B:1, C:1	A:1, B:4, C:0, D:1	1.0	0.0	0.86605	0.49500
4	A:4, B:1, C:1	A:3, B:2, D:1	1.0	0.0	0.06488	0.49500
2	A:4, B:1, C:1	A:1, B:4, C:0, D:1	1.0	0.0	0.04803	0.49500
1	A:4, B:1, C:1	A:5, B:0, C:1, D:0	1.0	0.0	0.01203	0.49500
1	A:4, B:1, C:1	A:0, B:4, C:1, D:1	1.0	0.0	0.00901	0.49500

Table 4.16: Position D

Min Payoff	Proposal1	Proposal2	Prob1	Prob2	Proportion	Fitness
1	A:4, B:1, D:1	A:4, B:1, D:1	0.4	0.6	0.40166	0.49500
1	A:4, B:1, D:1	A:4, B:1, D:1	0.2	0.8	0.31420	0.49500
1	A:4, B:1, D:1	A:4, B:1, D:1	0.3	0.7	0.23461	0.49500
2	A:4, B:1, D:1	A:4, B:1, D:1	0.4	0.6	0.04352	0.49500
1	A:4, B:1, D:1	A:4, B:1, D:1	0.1	0.9	0.00600	0.49500

Observe that no proposals from positions A or B are accepted, but that the proposals from positions C and D are immediately accepted. This phenomenon is observed in almost all simulations, with position C always proposing the coalition (A, B, C), and position D the coalition (A, B, D). The positions C and D always proposes to give themselves 1 of the payoff, with A and B getting the rest. All possible splits between A and B of the remaining 5 payoff are observed. Thus (A, B, C) and (A, B, D) gets accepted approximately 1/2 of the games each, with positions C and D both receiving approximately 0.5 fitness. Fitness for positions A and B take on values between 1 and 4.

A plausible explanation for the phenomenon of simulations resulting in proposals from A and B that never are accepted, is that since players A and B is a part of every winning coalition, it is not as important for them to have their proposals accepted. They can simply wait for players C and D to give proposals, since C and D have a great incentive to form proposals that will get accepted with themselves included. Thus it is less important what players A and B proposes, and more important for them to have a minimum payoff that is as large as possible, but small enough such that they accept the proposals from C and D.

4.2.1 Comparison with theory

Cooperative game theory

For this version of the game the core is not empty, but still no accepted proposals are in the core. There is a reasonable explanation for this. I believe that the reason is the proposals are limited to integer payoff splits in my model. There is no incentive for players C or D to accept proposals that give them 0 payoff, since this would give them the same fitness as not being part of any coalition. Remember that the reason for players C and D to receive zero payoff in the core, is that they are interchangeable and only one of them is needed to form a winning coalition. This leads to a race to the bottom where C and D will try to accept a little less payoff than the other in order to be a part of the winning coalition. Since the smallest non zero payoff is 1 in this model, this is where they end up. If real numbers where allowed I believe that the race to bottom would continue and the players would try to get less and less payoff for every iteration of the simulation. Their payoffs would get closer and closer to zero but never exactly zero.

The Shapley value is not observed in any accepted proposals for the same reasons as in the simple majority version. I averaged final fitness values of 30 simulations for the top strategies of each position, as I did for the simple majority version. The result was

$$(2.19685, 2.75991, 0.49972, 0.53202) \tag{4.2}$$

for the positions (A, B, C, D), which does resemble the Shapley value $(2.5, 2.5, 0.5, 0.5)$. I believe that the average fitness values for the interchangeable positions would get even closer as the number of simulations is increased.

Non-cooperative game theory

I used the same method as for the simple majority version in order to check whether the resulting prevalent strategies forms a Nash equilibrium, for the simulation that produced the plot in figure 4.4. This time I found that all prevalent strategies where best response strategies to each other. Thus the prevalent strategies for each position does form a Nash equilibrium, and we can predict that the observed stability will last in the long term.

5

Conclusion

5.1 Summary

During the course of this thesis I have studied different aspects of game theory, modelled and built an evolutionary simulation for a voting game, and analyzed results from running the simulation.

In the analysis I focused on two different versions of the game, with simple and greater majority requirements. In all simulations a form of cooperation occurred, where strategies aligned their minimum payoffs to other positions proposals and vice versa. Even though mixed strategies were allowed, strategies eventually converged to playing pure strategies.

Stable states that lasted until the end of the simulation were found in the majority of simulations. Some of those states could be guaranteed to continue by noting that the prevalent strategies for each population formed a Nash equilibrium. Other stable states could still last a long time when a very specific mutation was needed to break the stability, since the strategy space is large.

None of the described cooperative solution concepts could be seen in proposals. However when averaging the final fitness values for the top strategies of each position, a vector that resembled the Shapley value emerged. The resemblance was stronger in the greater majority version than in the simple majority version of the game.

5.2 Future research

The model has many limitations. One limitation is that only integer values are allowed in proposals and minimum payoffs. Another is the limited granularity of proposal probabilities. Removing some of the limitations and expanding the model could be interesting. The bargaining process could be made more complex in order to better fit some real world scenario. The minimum payoff could be changed to an arbitrarily complex voting function that determines whether to accept a proposal or not.

It would be interesting to build a similar, but probably more complex model, tailored to some real world scenario in order to explore if such a model could be used to predict behaviour in the real world. In many relevant real world scenarios the players do not only concern themselves with how much of the available payoff they receive. In political elections the ideologies of the parties is also factor. This could be modelled by letting players have more complex payoff and voting functions that take both coalitions and payoff splits as input.

Bibliography

[1] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. “Cooperative game theory: Basic concepts and computational challenges”. In: *IEEE Intelligent Systems* 27.3 (2012), pp. 86–90.

[2] Anders Eriksson and Kristian Lindgren. “Cooperation driven by mutations in multi-person Prisoner’s Dilemma”. In: *Journal of theoretical biology* 232.3 (2005), pp. 399–409.

[3] Kevin Leyton-Brown and Yoav Shoham. *Essentials of Game Theory: A Concise, Multidisciplinary Introduction.* 1st. Morgan and Claypool Publishers, 2008.

[4] Kristian Lindgren and Mats G. Nordahl. “Evolutionary dynamics of spatial games”. In: *Physica D: Nonlinear Phenomena* 75.1 (1994), pp. 292–309. ISSN: 0167-2789. DOI: `https://doi.org/10.1016/0167-2789(94)90289-5`. URL: `http://www.sciencedirect.com/science/article/pii/0167278994902895`.

[5] Daniel L McFadden. *The new science of pleasure.* Tech. rep. National Bureau of Economic Research, 2013.

[6] John Nash. “Non-cooperative games”. In: *Annals of mathematics* (1951), pp. 286–295.

[7] Heinrich Harald Nax. “Evolutionary cooperative games”. PhD thesis. Oxford University, 2011.

[8] J Maynard Smith and George R Price. “The logic of animal conflict”. In: *Nature* 246.5427 (1973), p. 15.