School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Internet: office@ecs.vuw.ac.nz

Decision-Making in One-Shot Games: The Emergence of Cooperation Through Pregame Signaling

William Kilty

Supervisors: Marcus Frean and Stephen Marsland

October 24, 2023

Submitted in partial fulfilment of the requirements for
Bachelor of Science with Honours in Artificial Intelligence.

Abstract

In our daily lives, decision-making processes frequently involve evaluating the trade-offs between personal interests and potential consequences. This is especially prevalent in decisions involving multiple decision-makers. The level of understanding between the individuals involved can significantly impact these decisions. In this paper, we investigate through machine learning the influence of understanding on decision-making within the context of interactions between strangers and individuals with varying degrees of familiarity and influence. We frame decision-making scenarios as one-shot games played between strangers without prior interactions, identifying the limitations in personal gain when dealing with unknown entities. Our study explores how knowledge of opponent behaviours can emerge through a value iterator-driven 'pregame', enabling the development of trust between actors in a digital environment. Focusing on classic game theory dilemmas, such as the Prisoner's Dilemma and the Stag Hunt, we illustrate that, even without a programmed concept of trust, actors learn to mitigate decision-making risks by engaging in a short pregame signalling act. This approach leads to the emergence of cooperation in situations where it would otherwise be considered perilous. Our findings shed light on the significance of knowledge and communication in decision-making and offer valuable insights for the design of rational actors in machine learning models engaged in one-shot game interactions.

Acknowledgments

I would like to thank my supervisors Marcus Frean and Stephen Marsland for their support throughout this project, without the help of which this would never have been possible. Thanks also to Victoria University of Wellington and its staff for allowing me the opportunity to pursue my interests in a formal setting and for teaching me all I know about computers.

Contents

Figures

Chapter 1

Introduction

At the heart of many activities in our lives are one-off decisions. ”Should I go for a walk?”, ”Should I have a slice of cake?”. These decisions are typically made through careful consideration of the benefits and consequences of these actions - eating cake is an enjoyable experience but may lead to health complications down the road. Some decisions, however, are made involving one or more other people. Often these people are friends - but sometimes these people are strangers, and on some occasions, we aren't able to even see the other person or people involved - if we're deciding whether or not to go to war, for instance.

The extent to which this lack of understanding of another person affects one's decision-making is a topic typically explored through the guise of philosophy. A topic unexplored as of yet is whether we can show or even quantify the importance of knowledge of others for decision-making within a machine learning model. In this paper, we study this idea, testing the limits of trust and observing the change in the quality of decisions made between strangers and between people we know and maybe even have some influence over.

Understanding the motives of those involved in a decision allows for better choices through interactions. The question thus boils down to: just what problems does a lack of understanding give rise to, and how do we overcome these issues in an interaction between strangers? The answer ultimately lies within game theory. We posit here that representing a series of possible decisions as one-shot 'games', occurring once between strangers with no prior interactions, will divulge an answer. The aim of this study is to investigate these one-shot games, discover by what means we can allow knowledge of opponent behaviours to emerge, and ultimately generate rational decision-making actors who can play a series of games that mimic these real-life interactions.

On their own, these one-shot games provide neither player with any context with which to determine how the other player(s) will act. However, with the use of a value iterator to learn an arbitrary-length pregame, we are able to build trust between actors. Primarily through the analysis of the Prisoner's Dilemma and the Stag Hunt, without telling the actors directly to trust one another, and instead motivating them purely through selfish means, we demonstrate that by the inclusion of a short pregame signalling act between the actors, they can gather enough information to mitigate the majority of the risk in their decision-making process - often leading to the emergence of cooperation where it would otherwise be dangerous to consider.

Chapter 2

Background

2.1 Game Theory

The concept of Game Theory exists to explain the interactions between rational actors in a mathematical way. Many situations in life involve interactions between people making optimal choices based on preferences derived from knowledge. These situations can be explained as games played by 'actors'. Actors are given a set of actions and a set of preferences over those actions. The interactions therein are known as a 'game'. There are few requirements for a situation to be explained as a game - actors must each choose one action from the total set of actions at a time, and their preferences must be consistent. [8, 10, 11] As such, game theory is a vast topic encompassing a limitless number of scenarios, and is utilised by a wide variety of fields. These fields include economics, biology, politics, philosophy, and computer science.

To discuss a game's makeup more formally, we can assign some terminology. In a single-actor game, we could show the preference for an action a from the set of actions A as $u(a)$. If an actor prefers action a over action b, we would say $u(a) > u(b)$. The preferences of actors are often given a value. While in standard games these values determine the strength of the preference, in ordinal games the actual values of these preferences do not matter; so long as the order of preference holds. We could, for instance, assign $u(a) = 1$ and $u(b) = 0$, or we could assign $u(a) = 100$ and $u(b) = 5$. These would be equivalent. [10]

In a two-actor game, a single action no longer holds a direct preference. It is the pair of actions (a, b), $a, b \in A$ that make up a 'state' $s = (a, b)$ in a game - and we would likewise assign preferences for an actor x - from the set of actors P - to each state $u_x(s) = u_x(a, b)$. The development of preferences over these states is called learning a strategy. If these preferences and the resulting strategy of both actors are the same for their respective actions, the game is called symmetric. Once a strategy has been found, we have identified preferences over each state, and our actors can play the game.

'Playing' a game can take several forms. Sequential games offer actors turns that run until a set end-state has been reached, where typically 'winners' and 'losers' are declared. Simultaneous games deal rewards to actors after each has made an action simultaneously. Within Simultaneous games, there are two subsets: repeated and one-shot games. Repeated games recur indefinitely to test long-term strategies without clear 'winning' conditions. One-shot games, the likes of which we will be studying here, are those in which a single set of actions is made without communication and each actor tries to maximise their score for this single round. This single action is now the actor's strategy.

2.2 One-Shot Games and Nash Equilibria

While various forms of continuous games allow for clever, cooperative strategies to develop over time - in fact, often the longer games run the more cooperation seems to develop [5] - One-shot games almost exclusively lead to greedy strategies where each actor selfishly tries to maximise their score, sometimes to the detriment of both actors. An example of this is the Prisoner's Dilemma (PD). This is a one-shot game in which two actors - taking the role of prisoners - are talked to independently after being caught for a crime. If one of them is to pin the blame for the crime onto the other, he or she is let go and their counterpart serves the full sentence. However, if *both* attempt to pin the crime on the other, they have the full sentence divided between them. If neither defers any blame, they serve a reduced sentence. We can define this as an ordinal game theory problem where actor x can either cooperate c with or defect d from actor y in which the order of preference for the states is as follows:

$$u_x(c_x, d_y) < u_x(d_x, d_y) < u_x(c_x, c_y) < u_x(d_x, c_y)$$

Looking at this with no knowledge of what y will choose, we can observe that the worst result is obtained from x cooperating, and the best result is obtained from x defecting. The strategy becomes clear that one should always defect. Even if we know what y picks, we will always maximise our utility by defecting. Of course, both actors have this knowledge, and so both choose to defect, netting the second-worst score possible for each. How do we solve this?

To unpack this problem, we must discuss this type of outcome more formally. An outcome in which neither actor can improve their score by unilaterally changing the state of the game - much like the d_x, d_y state of PD, where if either chooses to cooperate, their punishment grows - is known as a Nash Equilibrium. This kind of strategy, while always considered a solution to a one-shot game, is not necessarily optimal. Indeed the state in which both actors in the PD cooperate c_x, c_y provides a higher reward for both participants than d_x, d_y - though this is, itself, not a Nash Equilibrium, as either actor changing their strategy to defect from c_x, c_y yields a better reward.

Another game of interest in regards to Nash Equilibria is the Stag Hunt (SH). This is a one-shot game of two actors who wish to go hunting. If they work together, they can successfully hunt a stag, each taking home a large quantity of food. They are alternatively able to hunt hares which provide a lesser amount of food than their share of the stag. However, to catch a hare, one needn't work with the other hunter. If one chooses to hunt the stag, and the other hunts a hare, the stag hunter comes home with nothing. Here we get a different order of preferences than PD for actor x in an ordinal scenario:

$$u_x(c_x, d_y) < u_x(d_x, d_y) \geq u_x(d_x, c_y) < u_x(c_x, c_y)$$

In this scenario, we have *two* Nash Equilibria. If both actors defect (hunt hares), then one choosing to cooperate instead (hunt the stag) will result in a loss of reward. Similarly, if both are cooperating, then the change of strategy to defect will net that actor a lesser reward. This pair of equilibria is, however, unequal. Both defecting nets a worse reward for each than both cooperating. So while both are 'solutions' to the problem; one is optimal and one is not. Now the question follows; how do we encourage actors stuck in the lesser equilibrium to switch to the optimal?

These equilibria are common phenomena throughout game theory, and while continuous games allow actors to develop long-term strategies such as 'tit-for-tat' to escape from Nash Equilibria, such an equivalent does not exist for one-shot games. The primary limitation appears to be a lack of communication. In continuous games, though direct communication may not be possible, it *is* possible to glean information about one's opponent

through their choice of actions as the game continues. This basic form of communication through signalling actions could theoretically, if applied to one-shot games in the form of a 'pregame' display, provide a pathway out of a non-optimal Nash Equilibrium.

This pregame must have an endpoint - we need to get to the one-shot game itself at some point. The pregame therefore cannot be continuous. Equally, we do not want to force a fixed end-point after a given number of iterations, as the actors may not have decided upon a valid strategy in the time allotted. Rather, we should give the actors themselves the power to end the pregame as and when they choose. At its most basic, these pregames will take the form of an action set consisting of all the strategies for an actor within the one-shot game, the ability to not take an action, and the ability to 'leave' the pregame, initiating the one-shot with whatever strategies the actors are employing at the time of inception. The other possibility is the inclusion of an affordance to the actors in some form or another to influence the actions of their opponent more directly.

So, to conclude, we have devised a means for actors in a one-shot game to communicate in a sequential manner through the form of a pregame, where they publicly decide on the strategy they wish to employ within the actual game before it takes place. But how to enact such a means?

2.3 Value Iteration

Value iterators are machine learning tools that iteratively learn a policy π to navigate a Markov Decision Process [3]. A Markov Decision Process (MDP) is a system of states S and actions A which allow the transition from a given state s to a new state s' via action a with probability $T(s, a, s')$, supplying a reward upon arrival of $R(s, a, s')$.[9] These four elements; S; A; T; and R make up a representation of an environment and the relationships between states in that environment.

Our pregame is already a system of actions. A state is, as discussed before, simply a set of the different actions chosen by our actors at a given moment. We can take T as a binary system where performing an action has a 100% chance of altering the state to represent the performed act, and a 0% chance of reaching any other state. Our reward matrix R is simply 0 unless we perform the 'leave' action, in which actors are given the reward of the result of the one-shot game.

With an MDP, we develop π by calculating a numeric value for each action from each state Q. This is done via the Bellman Equation as follows:

$$Q\left(s, a\right) = \sum_{s' \in S} T\left(s, a, s'\right) \cdot \left(R\left(s, a, s'\right) + \gamma \cdot V\left(s'\right)\right)$$

Where γ is a discount factor ≤ 1 to prioritise shorter-term rewards over longer-term ones - to encourage our actors to leave the pregame in a timely manner - and where V is the calculated value of each state - 0 to start with. This equation values an action from a state as the sum of all the discounted values of all the other states and the reward of ending up in each state from here scaled by the probability that we end up in that state in the first place. From each $Q(s, a)$ value, we can determine the value of each of the states $V(s)$.

$$V\left(s\right) = \max_{a}\left(Q\left(s, a\right)\right)$$

The value of a state s is simply the value of performing the action valued highest from s. If this process is repeated, performing Q steps and V steps in sequence, the value iterator will converge on optimal values for each action at each state $Q*(s, a)$. These values make our policy π, where we aim to choose the best action at each state, maximising our rewards.

2.4 Related Work

Other papers have looked into game theory solutions through AI in the past. However, none apply the concept of a pregame encounter to determine trust within an eventual one-shot game. Perhaps the closest example to this concept is found in the paper 'Learning in One-Shot Strategic Form Games' by Altman et al. [1]. This paper encompasses a machine-learning approach to learning the strategies of opponents before a one-shot game. This is done not through communication between participants as we investigate but through prior history in other one-shot games. This is not to be confused with continuous games, in which the same game is repeated between the same pair of actors ad-nauseam; this paper explores the possibility of actors playing strategies in completely separate games with different opponents revealing insights into how an actor may approach an unseen game, thus allowing actors to choose a strategy to best maximise their score.

Experiments for Altman et al.'s paper were done through university students supplied with a set of possible strategies for a set of varying one-shot games. They were instructed to pick a strategy for each game, and these strategies would be faced against the strategies chosen by other students for the chance at a prize. Of course, the competition was not the focus of the study, but rather to supply the researchers with a dataset of strategies for games with which to train a set of association rules to predict strategies for unseen games.

They were able to predict using these rules with greater than 50% accuracy, the chosen strategy, in almost all played games. Furthermore, they determined these rules were marginally (3%) more successful than simply predicting the most common strategy of that player in the past. The paper concludes that though simple association rules are only marginally better than naive predictions, there is potential for more sophisticated Machine Learning methods to better learn associations and allow actors to better maximise their scores against unseen opponents in a new one-shot game.

The most significant inspiration for this work is the paper 'Holds enable one-shot reciprocal exchange' by Frean and Marsland [7]. This paper, though not directly discussing game theory, uses a value iterator to explore the origins of trading in societies through a game. They state that during an exchange, humans assume a certain risk of their trade partner simply stealing their trade item(s) and keeping everything for themselves. They therefore posit that for such a high-risk event to take such a central role in society, there must be an affordance that removes or reduces such risk.

A game is established in which two actors have possession of an item. They each value the other's item more than their own, though they still value their own more than having nothing. The actors wish to maximise the value in their possession and can leave the encounter whenever they wish. Ideally, an actor would leave with both items, resulting in the largest score. In this basic form, they established a value iterator to learn the best policy to navigate the exchange, and both actors simply left with their own items.

Clearly, the actors determine that attempting to swap items is too much risk. So how to overcome this? By the advent of affordances - specifically: holds. A hold is the ability to 'take hold of' the other actor, restricting the ability of both to leave the encounter until the 'holder' decides to release their hold. Running the experiment again yielded a successful exchange! The actor with the first turn will take hold of the other, one by one they will release their items, and then take possession of the other's item before the first actor releases their hold and the pair exits.

Another form of holds deemed successful is the ability for both actors to hold the same item. Neither can leave if both hold the same item and so the same basic structure emerges. As for why neither actor attempts to take both items for themselves, the pair cleverly structure the encounter around never leaving oneself open for having both items held by their

opponent without an exit-limiting hold in place.

Finally, Frean and Marsland determine a form of holds deemed implicit holds, which act as a way for each actor to believe they will be restrained from leaving without the other's consent without any physical holds taking place. In this scenario, the pair will happily release their items and perform a successful swap before leaving. This, it was determined, is the system with which our current daily exchanges take place. The belief that one will be stopped from abusing the system allows a pair of traders to exchange goods without fear of such abuse.

The implications for the seeming ease of this process raise questions as to why implicit trading appears to be a human-only behaviour, though the fact that such a solution exists in the first place provides a thorough insight into how the foundations of our society were established.

An interesting note from this paper is the apparent relation between the item swappers and the Prisoner's Dilemma. If we observe the order of potential exit-state reward preferences, this should reveal itself. For actor x against actor y where c is having released one's own item in favour of the other, and d is retaining hold of one's own item, we obtain states; c_x, c_y - a successful swap; d_x, c_y - x leaving with both items; c_x, d_y - y leaving with both items; and d_x, d_y - no swap taking place. The order of preference for x here is as follows:

$$u_x(c_x, d_y) < u_x(d_x, d_y) < u_x(c_x, c_y) < u_x(d_x, c_y)$$

This is the same as in PD. This interesting relationship may be worth exploring.

Chapter 3

Methodology

Through our model we will explore a recreation of Frean and Marsland's [7] Item Swap game. This will validate our system against an existing method, thereby providing confidence that this artefact works as intended. We will also explore the Prisoner's Dilemma, both as an interesting problem with no optimal Nash Equilibrium and to investigate whether the potential relationship between itself and the Item Swap is anything more than superficial. Another game of interest worth exploring is the Stag Hunt. This game, as discussed in section 2.2 has two Nash Equilibria. Pre-game posturing may just provide a means to avoid the suboptimal equilibrium in favour of hunting a stag. Finally, we will broaden our horizons and explore the effects of the pregame on all 2x2 ordinal one-shot games to better see the effects of communication across a variety of situations.

3.1 The problem with Value Iteration

Value iteration and the games we will be looking at have a key difference that makes them incompatible. Value iteration in its purest form is only applicable to one-actor scenarios. If we take a moment to examine the process of value iteration more closely, this limitation should make itself clear.

Value Iterators see a representation of their environment as an MDP. An MDP, as discussed in 2.3, contains a set of states S, a set of possible actions A, a transition matrix T, and a reward matrix R. It is within A that the issue arises. If we take A as the set of all actions performable by all actors in the game, the value iterator will perform the action with the highest reward, regardless of the actor acting. In this scenario, R must either take some global fitness score - at odds with the very basis of competitive games - or apply to a specific actor - in which case this actor is learning to play moves as another if it so benefits.

A must therefore contain the actions of a single actor $p \in P$ where P is the set of both actors. This establishes the need for actor-specific MDPs, denoted MDP(p). Following this revelation, we require actor-specific value iterators to learn the strategy of a given actor p from their respective MDP(p). This, however, has its own complications.

The two value iterators are independent of one another. For two actors p and q, the values changed by p's value iterator - the actions in the action set of MDP(p) - are assumed static by q's Iterator. This means that when developing the policy for actor q, π_q, we are assuming that it is impossible for actor p to ever perform an action in its action set.[1] This assumption is clearly incorrect, as both actors will be performing actions within the pregame.

[1] An interesting note here, the value iterators in this situation will find any Nash Equilibria within a game! Refer to 2.2 for the definition of a Nash Equilibrium to see why.

3.2 Comprehensive Value Iteration

To solve this issue, I developed a technique dubbed Comprehensive Value Iteration (CVI). CVI takes a separate MDP for each actor performing actions p, against each actor as the 'recipient' of the rewards q. The calculated Q_q values are then denoted $Q_{p,q}$ for some $p \in P$. Our V_q values are then calculated as such:

$$V_q(s) = \frac{\sum_{p \in P} Q_{p,q}\left(s, \max_a\left(Q_{p,p}\right)\right)}{|P|}$$

This ensures that rather than assuming values that actor q has no control over are static, we take the value of a state s as the average of the resulting reward of each actor performing its own desired action from that state $\max_a\left(Q_{p,p}\right)$. This way we can acknowledge the possibility of values outside of q's control changing with equal weight to the values q's own action action-set allows.

With this dynamic system in place, the decision to limit to the standard two-actor versions of these games becomes arbitrary. In theory, we could introduce any number of actors into these scenarios as we like. However, we are limited by the computation power of the device this system is run on. In this chapter, we have grown the computational and memory complexity of the system with respect to the number of actors from $O(1)$ to $O(|P|^2)$. In order to take advantage of any system with a large number of actors, we need a simplification of CVI.

3.3 Symmetric Value Iteration

Another system I developed is Symmetric Value Iteration (SVI). This is a simplification of CVI, where we build in an assumption of use within a symmetric game. This allows us to build a single π for one actor q, and reuse it for each of the others. This still requires the CVI calculations for $Q_{p,q}$, $\forall p \in P$, but rather than a complexity of $O(|P|^2)$, we have $O(|P|)$ - a linear scale rather than quadratic. This is a drastic improvement and allows use in symmetric games with arbitrarily large actor counts without any loss in mathematical accuracy.

Chapter 4

Implementation

4.1 Phase One: Value Iterator

To create the artefact, we first need a working value iterator. In its first design, this took the form of a basic one-actor Iterator with Bellman equation Q and V calculations running in a loop for as many iterations as specified by the user. To fuel this, an MDP with precalculated γ value, number of states and actions, T matrix, and R matrix is supplied to the Iterator. After ensuring the system was working, and that a single-person Stag Hunt did indeed value hunting a stag for a bigger reward over hunting a hare for a lesser reward, we moved on to phase two, where we designed a system to allow for easy definition of a game - rather than requiring precalculated values.

4.2 Phase Two: OOMDP

This system would be deemed the Object-Oriented-Markov-Decision-Process (OOMDP). This system takes advantage of the object-oriented nature of Java, the language used to create the artefact for this very purpose. Rather than taking a precalculated transition and reward matrix, this system would devise them automatically. To do this we create individual Actor objects, give them a set of *boolean* values and a set of actions A to perform to alter these values, and deem any unique combination of the *boolean* values a state $s \in S$.

To save memory within the program an actor can be represented as an integer in binary form, with bits representing its values - e.g. 01 for an actor with two values, one set to *True* and one set to *False*. s can likewise be represented in this form - a joining of each of its actors' binary representations into a single integer - e.g. 10001101 for a game with four actors each with two values. This optimisation allows us to store an entire state - with all its complex rules and actors - into a single integer. A final state is needed - represented by a single bit of order one larger than its component actors make use of - the exit state. Because of the choice of ordering with these encoded state representations, we are able to find every state combination purely from this exit state. If one iterates through all the integers up to and including the encoded representation of the exit state, one will find they have explored every state in the game.

A reward matrix R is a three-dimensional array of rewards given for the act of starting in a state s, performing an action a, and ending up in a state s'. The transition matrix T is, likewise, a three-dimensional array containing probabilities that performing a from s results in s'. To generate these matrices, one needs to attempt all the actions in A from all the states in S to observe where in S, s' is.

An OOMDP must be created for every actor, as it is the result of actions being performed

by a single actor x and their associated probabilities and rewards. For instance, if x performs action a and the result is the toggling of bit x_i, this is a result unique to x. If actor y performs that same action a, they will not toggle x_i, but rather y_i.

An observed s-a-s' combination has its T value set to 1 (0 otherwise), and its R value set to the result of a predefined function taking into account the state of the underlying actors and their relation to the specified target actor. An example of this reward function for the Stag Hunt in this version of the program should make things clearer.

$$huntStag?state.allActorsCond((a) -> a.huntStag)?4 : 0 : 3;$$

Upon reaching the exit state, this reward function activates, giving a reward of 3 if a hare is being hunted, a reward of 0 if not all actors are hunting the stag, and a reward of 4 if everyone is hunting a stag together. This is the reward of a single actor upon reaching the exit state and will be assigned to R in the initialisation of that actor's OOMDP.

Once calculated, the OOMDP functions identically to the classical MDP. The state and actor objects are no longer needed, and the value iterator defined previously can accept it as such. If we declare a single Stag Hunt actor and generate its OOMDP and perform value iteration on it, we end up with the same result as when the MDP was precalculated. However, when we define two actors, we end up with two Nash Equilibria - two states in which the actors will gravitate towards and then leave from. This follows the theory on the Stag Hunt where if both are cooperative they will hunt a stag, but if one is defecting the other will also defect [12]. This is not our desired result - where the actors should learn to escape from suboptimal Nash Equilibria - and as such we need to explore the problem deeper.

4.3 Interface

As a short aside, purely looking at the printed output of the Q and V tables is not as informative as is preferable, and so I implemented a custom interactive interface with which to explore the game and its strategy as devised by the value iterator. To do this, I used Java Swing to provide a single JFrame - a window - with which we can draw directly onto. From this, a new Graphical User Interface (GUI) library is built which displays custom buttons on this screen that can be interacted with using the mouse or keyboard input.

In addition, I created visualisations of states - a ring, with the actors spaced evenly across, coloured by discrete steps in hue to separate the actors visually as clearly as possible. A character may be displayed atop an actor, representing its current values or simply as an identifier, depending on the use case. I wanted this system to be as malleable as possible when designing it, so any number of actors may be displayed, with any given visualisation of data within them.

Once a strategy has been found through value iteration, the actors can be interacted with to either play their best move according to the strategy or the user can 'play' as one of the actors to see how other actors react to a non-rational counterpart. This allows for a much greater understanding of the strategy developed and allows a user to learn the nuances of the observed game. Fig 4.1 shows an example of this interface for the Stag Hunt problem.

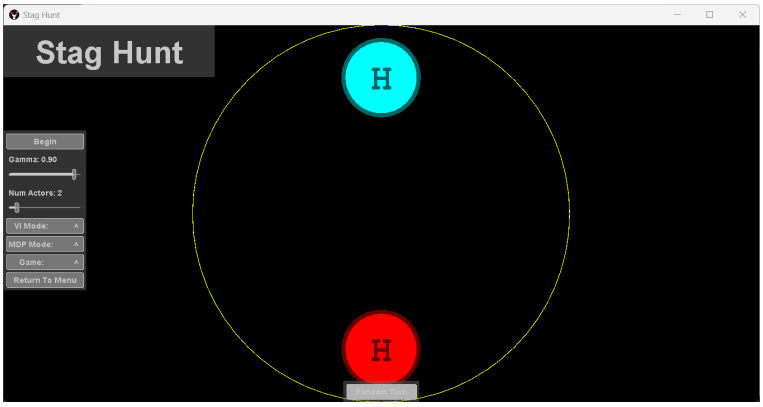

Figure 4.1: An example of the interface

4.4 Phase 3: Holds

In order to find a variant of our pregame system that allows for actors to learn to cooperate and escape from the suboptimal Equilibria, we implement the concept of 'holds'. This is the concept devised by Frean and Marsland [7] to enable convergence between their two actors in the game of Swapping. The idea is that an actor can hold the actor to its left - around the ring, as discussed in section 4.3 - to restrict the ability for both to 'leave'. This inclusion should remove any risk of being 'left' while an actor makes any risky moves towards a more optimal equilibrium in hopes that the other follows. An actor can hold its counterpart, switch to the desired strategy, and not release the hold until the other follows suit.

4.5 Comprehensive Value Iteration

Another vital element implemented within the engine was the development of Comprehensive value iteration (CVI). A limitation of standard value iteration is that values not influenced directly by actions within the OOMDP are assumed static. This assumption is problematic, as other actors within the game can - and likely will - alter these values. Rather than treating a value not directly controlled by the target actor as a fixed point to work around, we should instead attempt to predict the strategies of the other actors in the game and incorporate their strategies into our own. This is a concept discussed more generally through theoretical game theory by Fudenberg et al. in their 1991 book on game theory [8].

CVI was implemented by altering OOMDPs to take in both the actor performing the actions x, and the actor receiving the rewards y separately. When value iteration is performed, rather than taking in a single MDP for the actor, we take in an MDP for every actor as the target of every other actor's actions. After performing a set of Q steps for each actor, we can determine what a they each prefer for each s. This allows us, in the next step, to take the average of the valuation of all these 'best' s, a pairs to our target actor as the valuation for its next Q step. This way, if a state may allow x to achieve its best reward, but from that same state y is likely to incidentally give x its worst reward, x no longer naively assumes that it is guaranteed its best reward, and instead values the state as the average of these best and worst rewards.

4.6 Symmetric Value Iteration

A significant problem with CVI is that the computational and memory complexity balloons from $O(1)$ for the number of actors to $O(X^2)$. This puts a hard limit of 3 actors for a game of even moderate complexity. Whilst this allows us to perform non-symmetric games within the engine, Swapping, PD, and SH are all examples of symmetric games. To better explore these games, it is worth introducing another assumption to reduce this complexity. This assumption is that of symmetry, leading to the development of Symmetric Value Iteration (SVI).

SVI allows us to treat each target actor as identical. If actor x sees the state 01, in a game with two actors each of size 1, it will react the same way as actor y who sees the state 10. Note that this is not each actor acting the same in the *same* state - it is each actor acting the same in the *equivalent* state. To implement this, we only need an OOMDP for each actor acting against a single target actor. We are then able, after each Q, V step, to rotate the values for each state to their equivalent state for each of the actors, and simulate the effects of CVI without needing to repeat calculations unnecessarily. This is correct as long as we expect our actors to behave in the same way as one another - i.e. in a symmetric game. This state rotation is done through the line of code:

$$state = (state >> actorSize)|((state\&((1 << actorSize) - 1)) << (stateSize - actorSize));$$

This code uses bit manipulation to rotate the binary representation of a state by an actor, to reflect the equivalent state for the actor one space to the left. This process can be repeated to obtain an equivalent state for any given actor within a game.

Chapter 5

Results

Before beginning our foray into the results of the artefact, we must first discuss the meaning of the plots that will be shown. The primary output we will analyse is a representation of the pregame in what is called Extensive Form [8]. This is a decision tree-style output showing the progression from state to state until the conclusion of the game is reached - our actors decide to exit the scenario and enact the one-shot game. These plots are coloured through a specific colour scheme. A blue node in the tree is the start-point of a pregame. A grey node is a standard transition node of no key significance. A green node is one in which both actors agree to begin the one-shot. A yellow node is one in which at least one of the actors will decide to leave if given the chance. Each node forks into the outcomes of each actor playing its best-determined move from the state belonging to the node.

The other key visualisation of game output is Normal Form [8]. This shows the scores obtained by each actor for each state in a one-shot game in the form of a matrix. This is useful for displaying all the possible outcomes of a game, though it is cumbersome for more complex games or games with more than two players. A similar colour scheme is chosen for this representation. A grey tile is a state in which neither actor will choose this outcome through a pregame. A green tile is a state in which both actors will select this outcome if it were to occur during the pregame. A blue tile is one in which the blue player will choose should it occur in the pregame, and conversely, a red tile is one in which the red player favours.

5.1 Recreating Swaps

First, we recreate the Item Swap game proposed by Frean and Marsland [7] in a bid to show that the system works as intended. With standard value iteration, we obtain the extensive form tree as seen in fig 5.1.

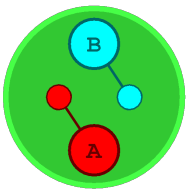

Figure 5.1: Item Swap with Naive Value Iteration

Unsurprisingly, standard VI does not enable actors enough communicative knowledge to enact a successful swap, and both actors elect to keep hold of their beginning items. Adding CVI to the scenario with no affordances of any kind gives us fig 5.2. This is still insufficient for a successful swap, though this is promising news as without affordances, Frean and Marsland were also unable to achieve a swap. It is only with the advent of some form of hold that a 'swap dance' is obtained. This can be seen through fig 5.3. Because this is a symmetric game, we unsurprisingly end up with a symmetric 'swap dance'. Either one of the actors x takes hold of the remaining actor y's item, thus locking the pair into the interaction. From here x is willing to release its own item and y is willing to take hold of x's item in any order. Once y is in possession of x's item, it will spend its next turn releasing its own item, thus releasing the lock between the two, and they may separate having successfully swapped.

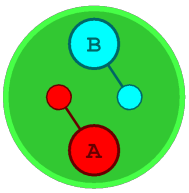

Figure 5.2: Item Swap with Comprehensive Value Iteration - no holds

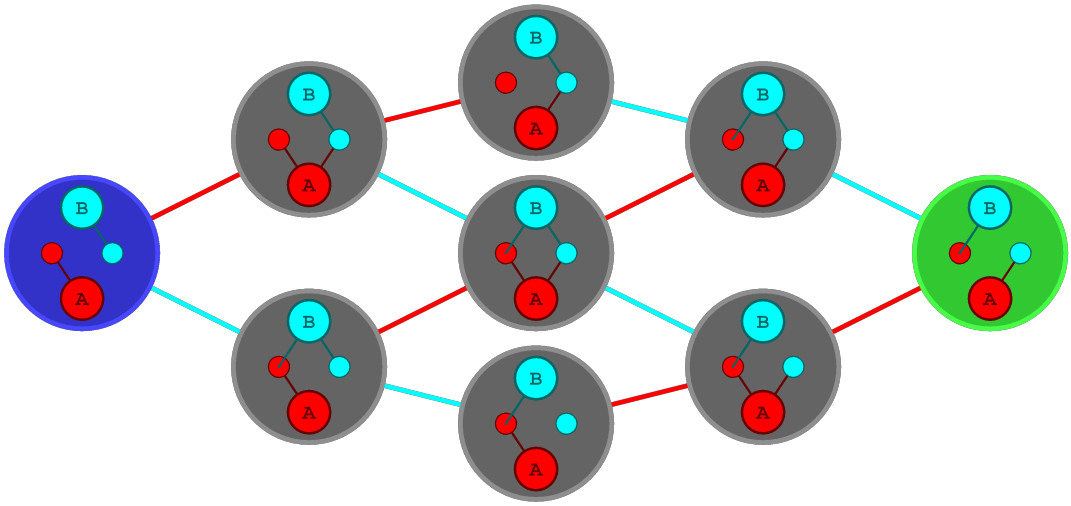

Figure 5.3: Item Swap with Comprehensive Value Iteration - with holds

We have shown here that under the same circumstances as Frean and Marsland's solution, we receive the same results. This validation ensures the model is working as intended, and we are able to experiment further. Namely, to try performing a swap between three actors. To make this scenario work with more than two actors, we must establish which items

are being swapped. An actor may hold either its own item or the item belonging to the actor to its right. Holding another actor refers to holding the actor to its left. This way an actor can specifically stop another from taking its own item should it have the need.

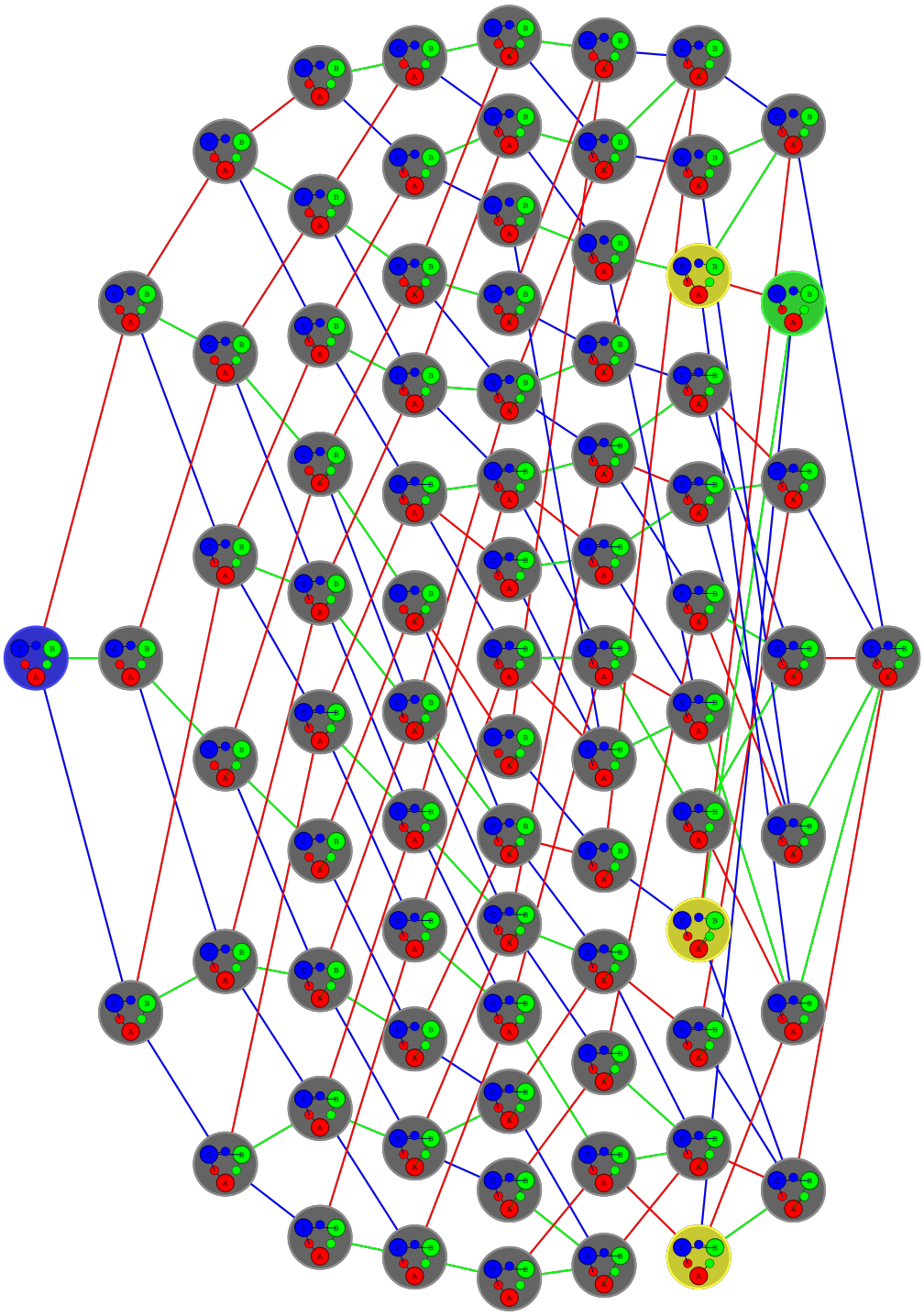

Figure 5.4: Item Swap with Symmetric Value Iteration - with holds and three actors

Running this experiment with three actors - fig 5.4 - provides a tree with three-way symmetry. The dance is vastly more complicated with three than with two. However, there are some interesting specific observations to be made from it. Perhaps most notable is the occurrence of yellow nodes in the network. Like the two-actor swap, we have a single green node in which the actors have all successfully swapped, but in order to reach this node, the actors must traverse one of three yellow points - points where one of the three actors will decide to leave, having already obtained its desired item and no restriction on its exit. This is unfortunately unavoidable in this scenario as actors are not able to restrict the exits of all participants on their own, and a restricted exit must be decomposed over multiple moves.

The other phenomenon present here is that while all holds were available in the two-actor swap, holding another actor directly was not utilised. For the swap dance to complete in the three-actor case, all types of holds are utilised to ensure risk is eliminated. This implies that performing a swap between more people introduces more risk. The reason for this can be observed within fig 5.4 where before releasing hold of their own item, an actor x will always take hold of the left actor y. This is to stop y from releasing its own item and making off with x's before x has obtained sole possession of its desired item.

5.2 Prisoner's Dilemma

Moving on from item swapping, we explore the effects of a pregame on the Prisoner's Dilemma. Starting with standard VI, we observe results in normal form in fig 5.5. This shows the dilemma as we hypothesised in section 2.2, with one Nash Equilibrium at d, d, and opportunistic choices for each at c, d and d, c. The actors will never agree to cooperate, for they could always hypothetically obtain a better score by defecting.

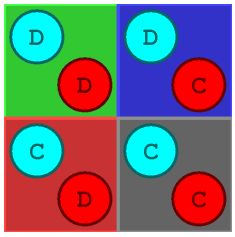

Figure 5.5: Prisoner's Dilemma with Naive Value Iteration in normal form

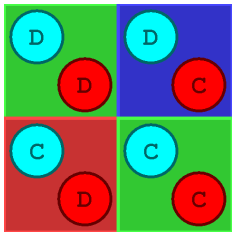

Figure 5.6: Prisoner's Dilemma with Comprehensive Value Iteration in normal form

When we introduce CVI/SVI (the two are interchangeable for symmetric games such as this), the actors suddenly agree to cooperate. This is seen in fig 5.6. However, it has not removed the opportunistic choices or the defect space as solutions. This can be seen across figs 5.7 and 5.8 where we have the pregame in extensive form. The actors in both cases simply leave immediately. To leave from d, d as a starting space, one needs to cooperate. This

cooperative signalling, however, provides the other actor an opportunity to defect under the knowledge that it will now get the ideal payout. This means the two will simply agree to take the suboptimal d, d payout. Conversely, they now realise that from c, c they can risk the better reward of d, c by changing and hoping for a chance to act before their opponent, though if their opponent acts first the pair get stuck once more in d, d. The risk is deemed too high, so the pair agree to cooperate instead.

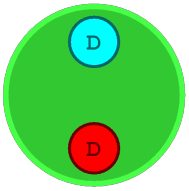

Figure 5.7: Prisoner's Dilemma with Comprehensive Value Iteration starting with defection

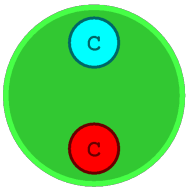

Figure 5.8: Prisoner's Dilemma with Comprehensive Value Iteration starting with cooperation

If we enable holds for the pair, we get a new range of results. The normal form can be seen in fig 5.9 whereas extensive forms starting from d, d and c, c can be found in figs 5.10 and 5.11 respectively.

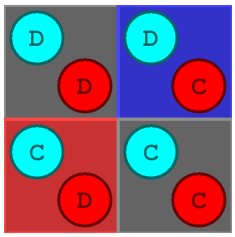

Figure 5.9: Prisoner's Dilemma with Comprehensive Value Iteration in normal form with holds enabled

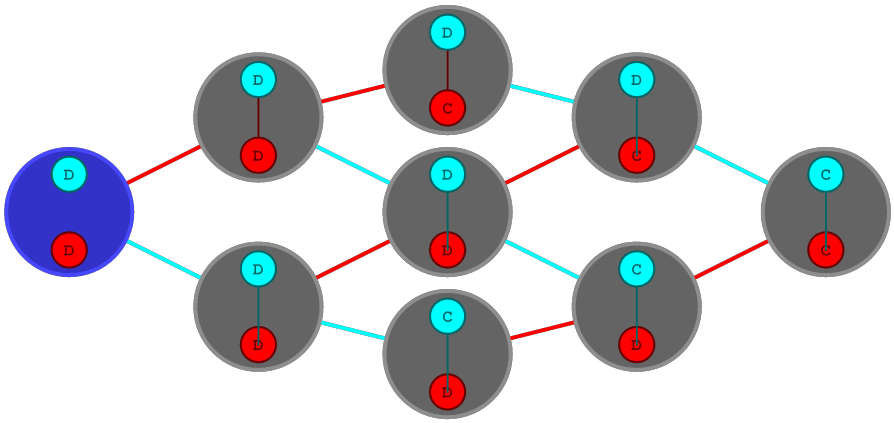

Figure 5.10: Prisoner's Dilemma with Comprehensive Value Iteration starting with defection and holds enabled

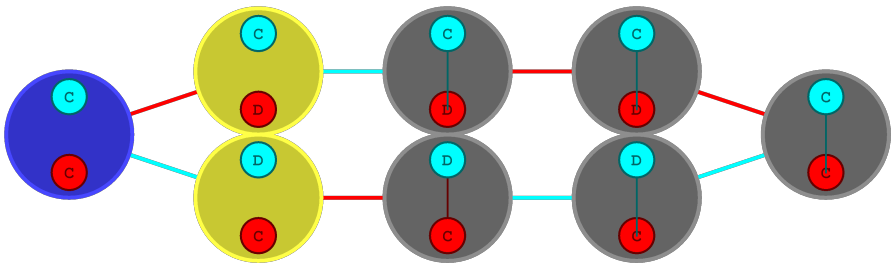

Figure 5.11: Prisoner's Dilemma with Comprehensive Value Iteration starting with cooperation and holds enabled

Interestingly, while holds enable a swap dance to function, they appear to completely break the Prisoner's Dilemma. Despite superficial similarities between Item Swap and Prisoner's Dilemma, the games appear to have a key difference. We shall discuss this in greater depth later. For now, we observe that holds have removed the option to exit through any path other than the greedy maximum. The actors will neither cooperate nor compromise for the split sentence. When starting from c, c, the first to act will have the opportunity to defect and betray the other. However, should their opponent take a turn, a hold will take place and the pair will never let go of one another. From the other start point - d, d - the pair will simply take hold and never let go, with no chance of exit at any point. A pregame with holds in place will result in a stalemate almost every single time.

Once we increase the number of actors beyond two, the actors will ignore the holds entirely, instead opting to make their way towards the total defect strategy, allowing chances for individual opportunity along the way. This can be seen for three actors and four actors in figs 5.12 and 5.13. Ignoring holds is simply because actor x holding actor y leaves actor z free to defect and leave while x and y are stuck in their hold. Moving towards the all-defect state is likely due to the increased opportunity to take a greedy result. When defecting, the

blame is put solely on those who are deciding to cooperate. For all actors to defect, there are a minimum of as many turns as there are actors to switch to defecting from all cooperating. This provides the first moving actor with $|P| - 1$ opportunities to have another turn and leave without punishment. With two actors, this is a big risk, giving a 50% chance of ending up in d, d. But with three actors the chance is only 33%. The actors deem the risk worthwhile and refuse to cooperate.

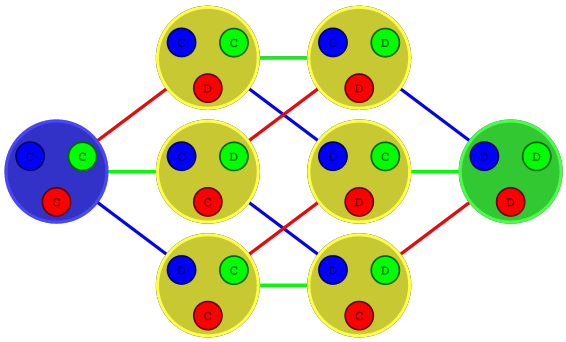

Figure 5.12: Prisoner's Dilemma with Comprehensive Value Iteration starting with cooperation and holds enabled - 3 actors

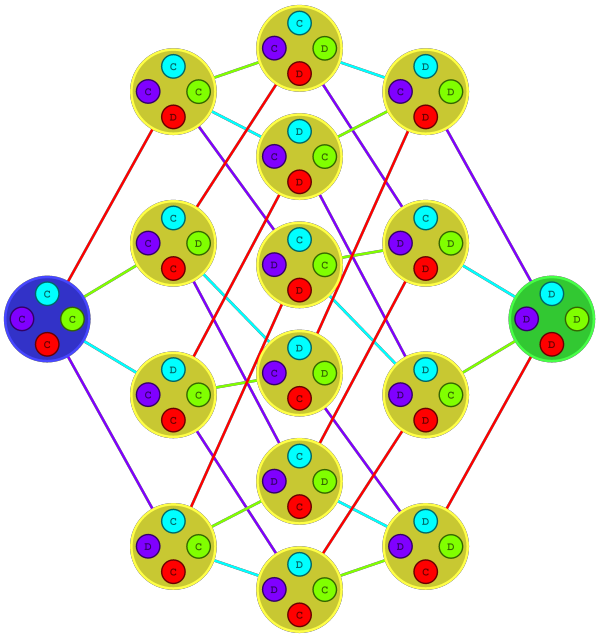

Figure 5.13: Prisoner's Dilemma with Comprehensive Value Iteration starting with cooperation and holds enabled - 4 actors

5.3 The Stag Hunt

For the Stag Hunt, running a standard value iterator gives us what we predicted in section 2.2 as seen in fig 5.14: a set of two Nash Equilibria; one for hunting stag as a pair; and one for both hunting hares. Upon introducing CVI, we have successfully eliminated h, h as an exit point. The actors will now elect to cooperate in any situation. This can be seen in the normal form - fig 5.15 - and through the extensive form - fig 5.16.

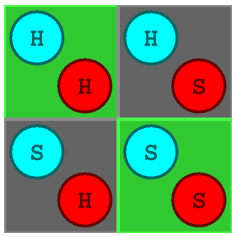

Figure 5.14: Stag Hunt with Naive Value Iteration in normal form, showing both Nash Equilibria as viable options

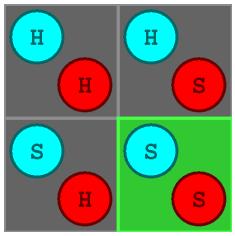

Figure 5.15: Stag Hunt with Comprehensive Value Iteration in normal form, showing stag hunting as the only chosen exit-point

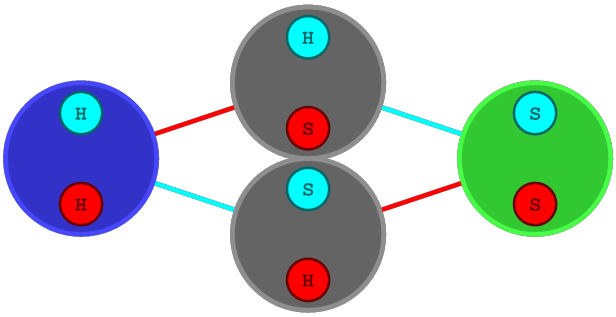

Figure 5.16: Stag Hunt with Comprehensive Value Iteration. Extensive form showing the process of agreeing to hunt a stag

If we add holds to the scenario, we receive the same plots. The actors do not need to restrict the ability of their opponent to leave to remove the risk of switching from defecting to cooperating. Simply the ability to signal effectively to one another, alongside the knowledge of what the other actor's preferences are, provides enough information for the actors to trust

one another. This is reinforced by Robert Aumann in his 1990 paper 'Nash equilibria are not self-enforcing' [2] in which he states 'In the games of [Stag Hunting], each player *always* prefers the other to play c, no matter what he himself plays. Therefore an agreement to play (c, c) conveys no information about what the players will do.' [pg.620] This means that if the actors understand the preferences of one another, they will both deduce that they both want to play c, and any interaction therein is ultimately unnecessary. It is the understanding of one another's preferences provided by CVI that allows an optimal Stag Hunt to occur.

Further, if we increase the number of actors in the scenario, we get the same results. This can be seen in the game of five actors in fig 5.17. The posturing takes longer as more actors need to take turns to signal a preference for stag hunting, but assuming all actors hold a preference for stag over hare, the stag hunt takes place.

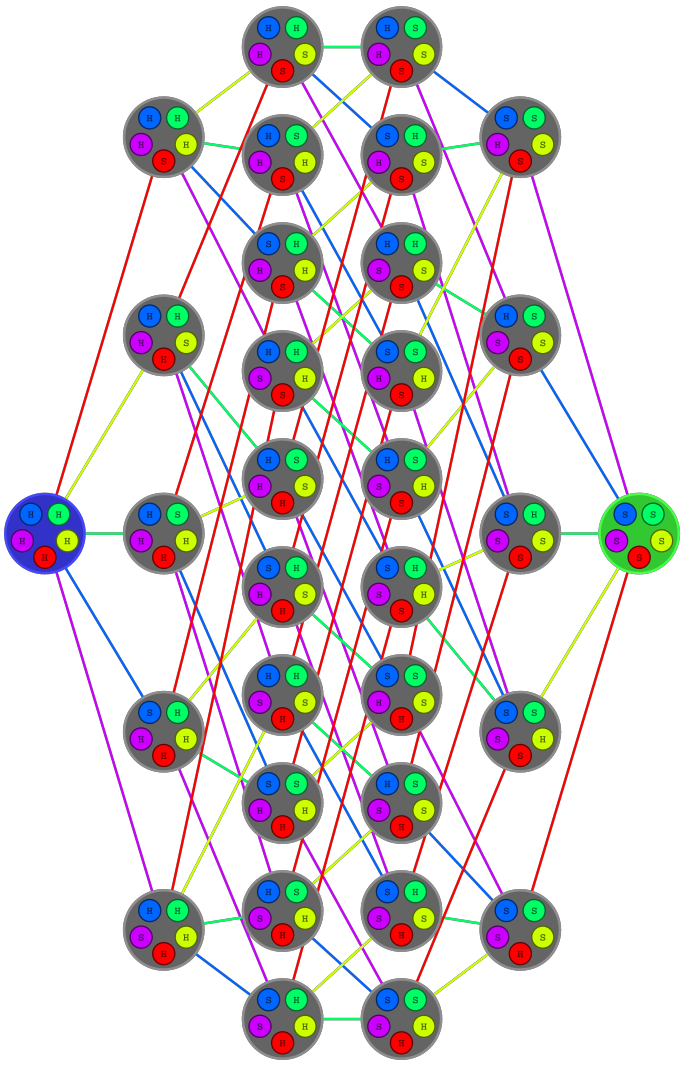

Figure 5.17: Stag Hunt with Comprehensive Value Iteration. Extensive form showing the process of agreeing to hunt a stag with 5 actors

5.4 All 2x2 Ordinal Games

Finally, we broaden our sights to the scope of all one-shot games. There are a total of 144 unique ordinal two-actor games where each actor has a single action. This encompasses both the Stag Hunt and the Prisoner's Dilemma as well as a number of other famous games such as 'Chicken' and 'Battle of the Sexes'. These games are obtained by taking all the non-rotational permutations of two actors with four different state preferences: $3! \cdot 4! = 144$.

These games are arranged in a grid, containing all the games in their normal form representations. To begin with, we show all the Nash Equilibria and personal optima in fig 5.18. Each 2x2 subgrid represents all the exit states of a game in normal form. The number within an actor represents the reward given to that actor playing that strategy. As these are ordinal games, the specific number itself is not significant, merely the order of preference $4 > 3 > 2 > 1$.

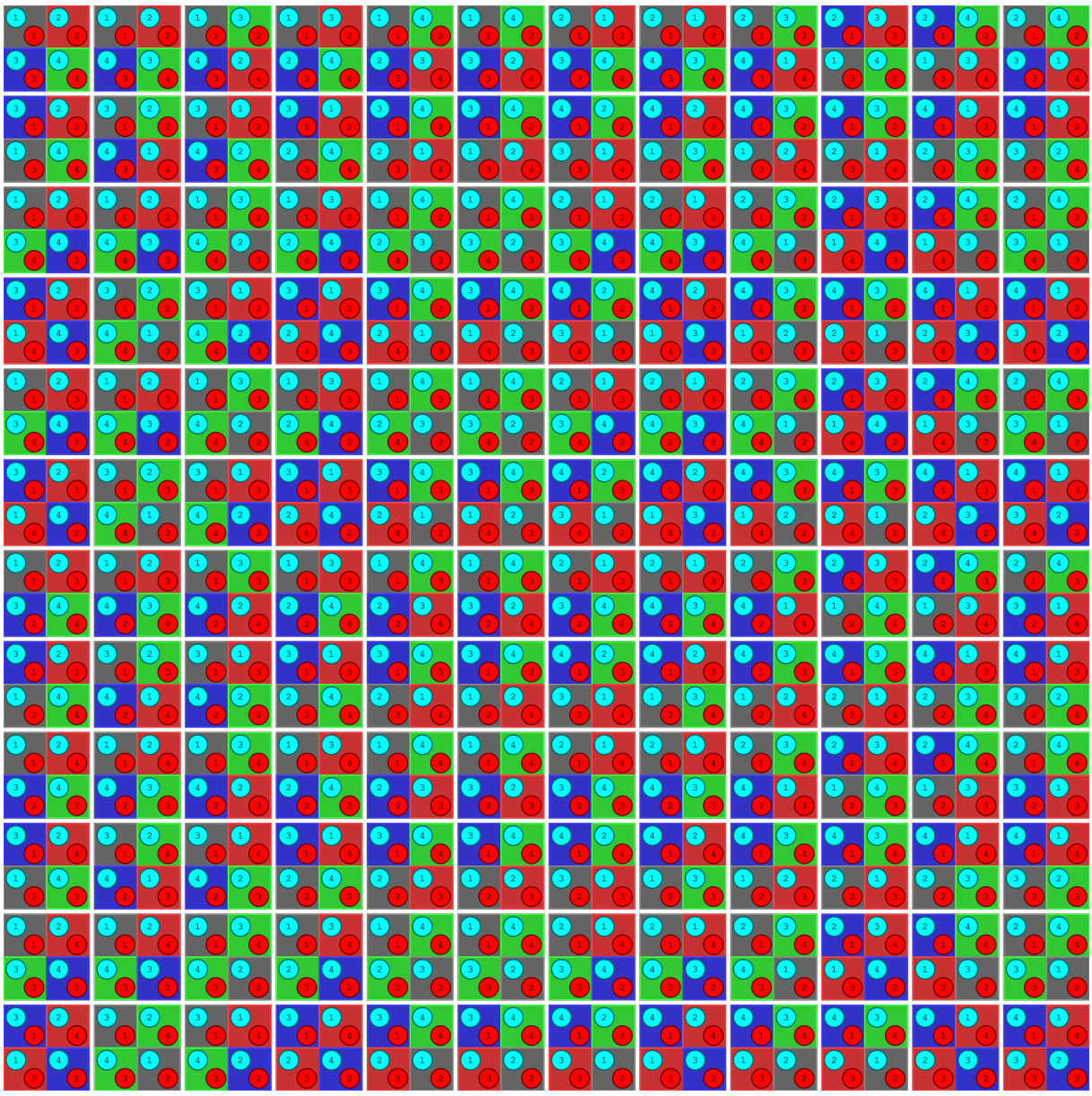

Figure 5.18: All 2x2 Ordinal Games. Nash equilibria in green. Personal optima in actor colours.

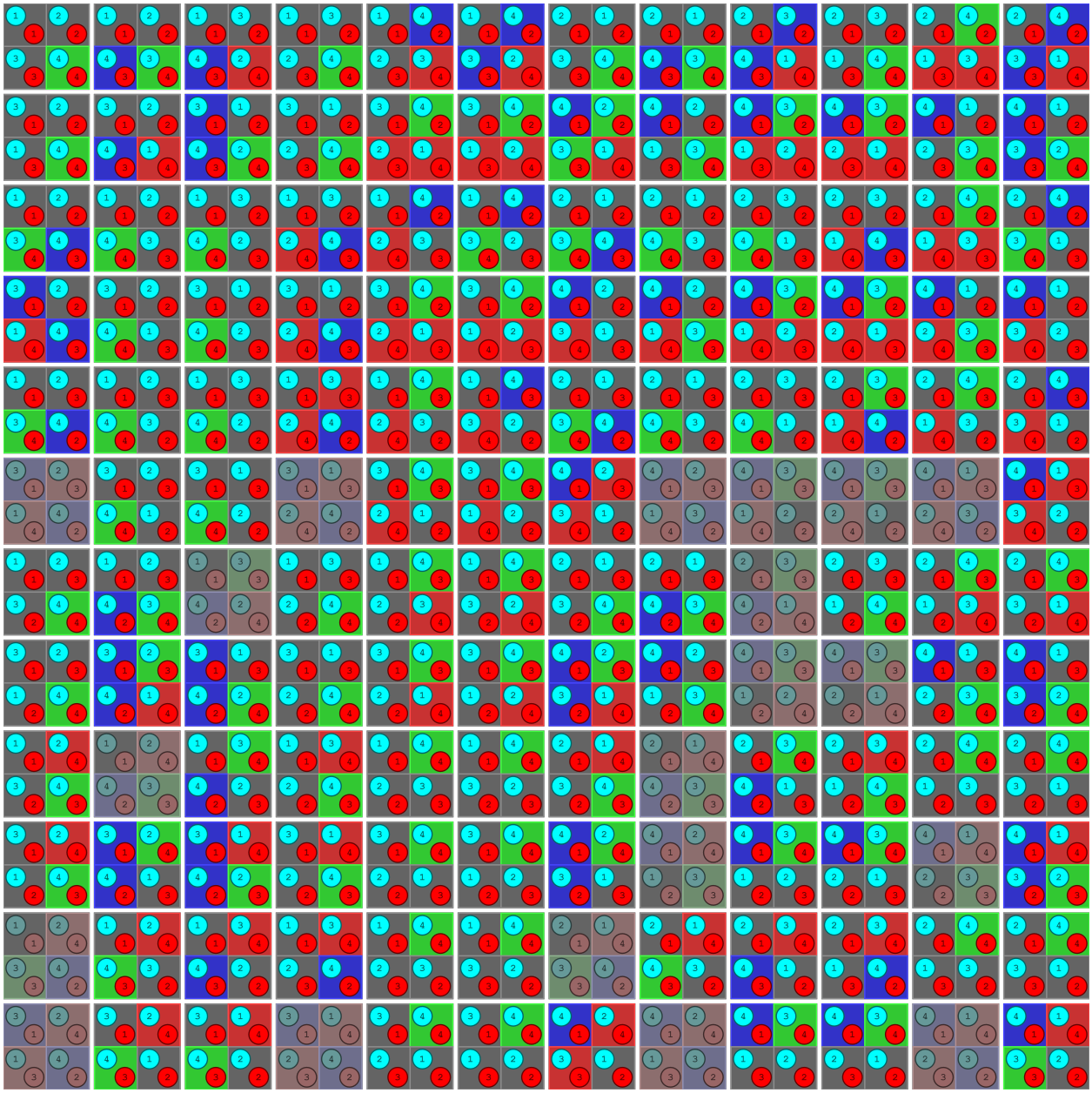

Figure 5.19: All 2x2 Ordinal Games under CVI. Cooperative solutions in green. Personal optima in actor colours.

Following this, fig 5.19 shows all games under the influence of CVI - allowing actors to communicate and understand one another's motivations. We have eliminated games that show no change under this addition. From this, it can be observed that the vast majority (86%) of games are altered through prior communication without affordances. The only games that are not affected are those that have goals so closely aligned as to make communication unnecessary and those that have goals so opposed that no cooperation is possible. Any game in which forward-thinking or some form of compromise is possible is improved through the addition of prior communication. Actors can escape from both suboptimal Nash Equilibria and suboptimal greedy solutions.

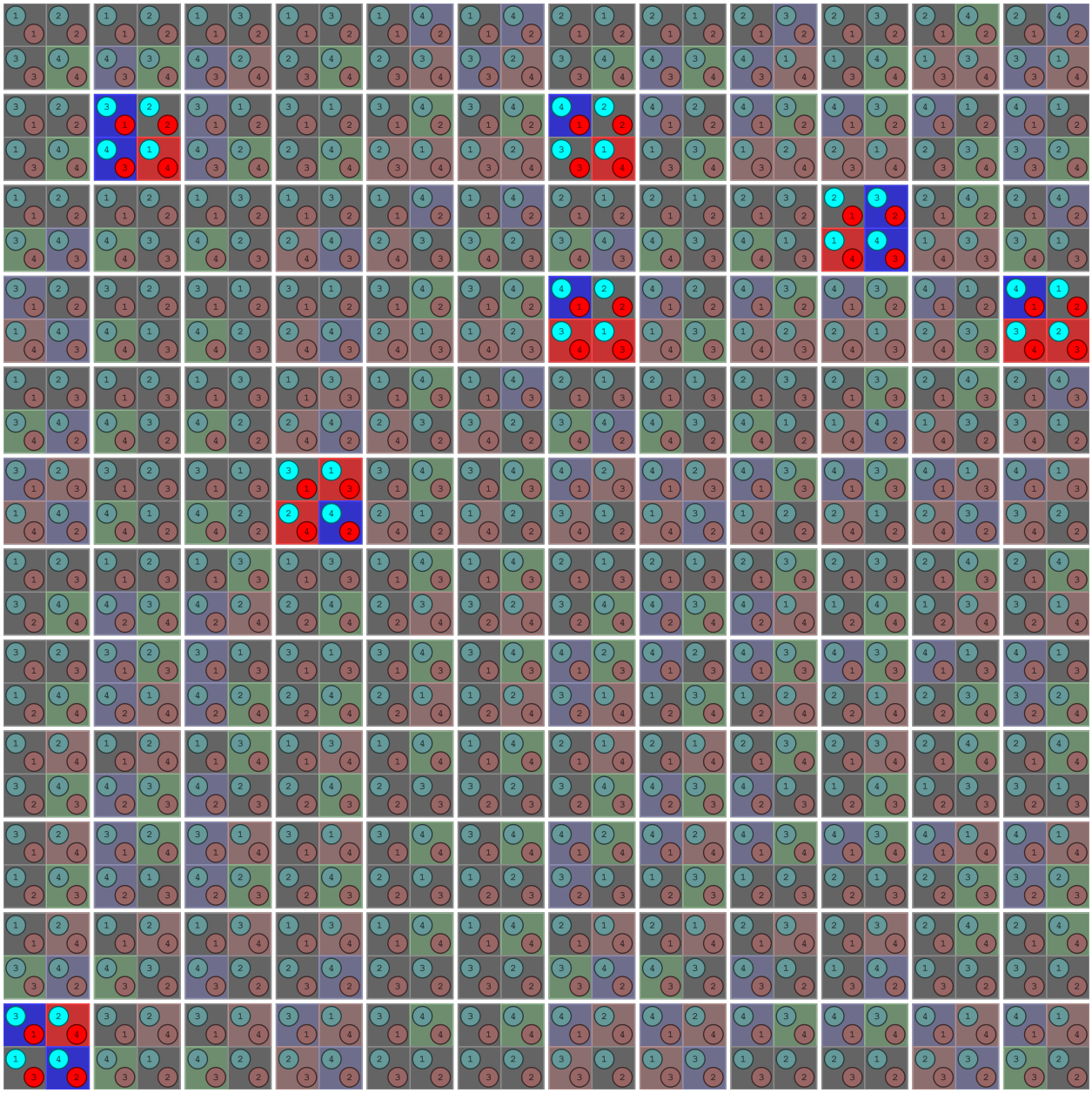

Figure 5.20: All 2x2 Ordinal Games under CVI with holds. Cooperative solutions in green. Personal optima in actor colours.

After introducing holds to the set of games - fig 5.20 - we eliminate all but seven of the total set - when compared to CVI. Holds generally do not influence the turnout of games when allowed within pregames. It appears that holds have an effect only when an opponent can be 'guided' towards an alternative solution that though slightly worse for the opponent, isn't enough of a detriment for the opponent to care. The only exception to this is the Prisoner's Dilemma, in which the game breaks down into a stalemate, as discussed in section 5.2. Interestingly, no other game matches the Prisoner's Dilemma in this way; where compromise is encouraged until the actors have direct influence over one another. This is not to say that holds remove compromise either - it is shown through item swapping that holds allow for a compromise that isn't possible without it, and many games still show compromise equilibria with holds enabled.

Chapter 6

Discussion

Pregames, through the use of CVI, allow actors to learn the preferences of their counterparts and communicate with them to escape from suboptimal Nash Equilibria in one-shot games. In the Item Swap game, the actors learn to swap in situations with more than two actors when holds are available. Interestingly, though holding actors specifically is not necessary for the two-actor scenario, with more than two, both holding other actors and locking items between actors becomes vital, as without either of these abilities, actors will refuse to swap. This is due to the increased complexity of the multi-actor scenario, where only having access to one's neighbours means there are elements of the environment one actor cannot control. Actors must work together to form chains in order to perform a swap, showing another form of cooperation emerging from these simple swap dances.

The similarities between Item Swaps and Prisoner's Dilemma are little more than superficial. While the exit states are essentially identical between the two games, in PD, actors have no ability to stop another actor from trying to take the greedy strategy. In Item Swapping, actors can simply hold onto their item, removing the ability for their opponent to try for both items. In this way, it is possible to transition from d, d to c, c without traversing directly through c, d or d, c. Moreover, reaching the d, c exit condition requires cooperation from both actors. One actor needs to consciously release their item without any holds in place, and the other needs to take it. This is not true for PD, where the difference between c, c and d, c is one move - dependent only on one actor.

Through CVI/SVI, Prisoner's Dilemma actors are able to compromise and agree to cooperate, despite the temptation to defect for a potentially greater reward. The benefits of cooperating are, however, not significant enough to encourage a transition from the defect-defect starting state. Indeed, with greater than two actors, the benefits are seen as less significant still, and actors choose to risk a defect strategy from any starting position. This may be indicative of the same pressures that lead to the 'bystander effect' [6] - the noted effect where in an emergency, if a large enough crowd is present nobody will call an ambulance. With more people present for an emergency, the pressure of being singled out as the person to call for help may be lessened. This may incentivise people to 'defect' by not spending the energy to engage with the situation, believing somebody else will 'take the fall' and cooperate thus avoiding the poorly valued 'all defect' state.

Looking at the Stag Hunt, we observe a sustainable shift from any starting state to the 'all cooperate' state when CVI/SVI is employed. This is true for all numbers of actors, given a large enough γ, of which the requirements increase for each subsequent actor in the scenario. This shows that the pregame provides enough of a means of communication to escape from extremely suboptimal Nash Equilibria. In fact, this means of communication is so successful at finding a 'best option' - when one is available - in decision-making that we can confidently say the pregame is an effective way of encouraging actors to cooperate. This

heavily implies that communication has a vital role in the process of decision-making in the wider world. People are far more likely to cooperate with one another towards a greater common goal if they can communicate effectively. The chance for miscommunications and incorrect assumptions decreases when actors are able to share knowledge.

We see from our results that pregame posturing has a beneficial outcome for almost every game involving two actors. This can be extrapolated to more actors through the exploration of Item Swapping, Stag Hunting, and the Prisoner's Dilemma as games of interest. It stands to reason that communication therefore plays a vital role in decision-making with more than one person. However, looking at the Item Swappers, we know that on some occasions communication is not enough to encourage an optimal outcome. An affordance in the form of holding - restricting the ability of an actor to leave the pregame - allows the actors to successfully transition to the cooperative optimum.

Interestingly, holds do not always improve the results in the same way communication appears to. In the majority of games, holds are found to be irrelevant to actors with sufficient knowledge of their opponent. In the case of the Prisoner's Dilemma, however, it can be argued that holds reduce the actors' performance within the game. The ability to hold one another results in a never-ending grapple where neither will compromise for anything less than their personal optimum - at the expense of their opponent, who won't let go until the tables are flipped the other way. This results in a stalemate. Even more interestingly, this phenomenon occurs even with the discounting γ value below 1.0 - even when short-term rewards are valued more highly than long-term ones. Beyond a certain point, the actors no longer consider options outside of their starting positions and thus accept whichever result they start with.

Looking back at the role of γ for the Stag Hunt problem, we achieve successful cooperation - switching to signalling for hunting stag rather than hares - so long as we allow the value iterator the ability to see far enough into the future for the benefits to outweigh greedy shortsightedness. This is affected directly by the chosen value of γ in the VI, as a value of 0.9 or less results in too great a focus on short-term rewards for actors to agree to cooperate. This is true to an extent for all game representations, where a small γ value will limit the ability to cooperate over long pregames. This is important to consider and has ramifications for the role of long-term thinking in cooperation. However, this paper does not delve into such a topic and leaves these ramifications open for analysis.

Another limitation of this study is the assumption of complete knowledge. Between standard value iteration and CVI/SVI, we have two extremes: complete lack of knowledge and complete knowledge. This is seldom the case in reality, and information is gleaned from people as we interact with them. Gaps in this knowledge are often filled with prejudices and stereotypes which can affect our level of trust in a person we otherwise do not know anything about. This topic is discussed in a related field by Burnett et al. in their paper 'Bootstrapping trust evaluations through stereotypes' [4]. However, a direct analysis with regard to one-shot game theory would still be insightful.

Chapter 7

Conclusions

In conclusion, we have developed a model capable of representing any one-shot game as the result of a series of sequential posturing steps between two or more actors. This dynamic system allows a thorough examination of the role of communication and understanding in decision-making. We have determined that both are necessary precautions in interacting effectively with others; to make the best possible decision available when others are involved, having knowledge of the people you are dealing with, and talking with them before making the decision is critical for both mutual and personal gain. Furthermore, enacting a form of control over another can net an individual with a greedy solution to a problem more beneficial to their interests. However, if both actors have equal power to directly influence one another, a discussion can break down, resulting in no decision being made - cooperative or otherwise.

Looking forward, a number of questions are left to answer around the topic. The exact role of long-term thinking in decision-making is still undefined. It is clear that short-term thinking leads to sub-optimal strategies in complicated scenarios, and that this is exacerbated by the addition of multiple actors. But the limits of this, the potential drawbacks of thinking too far ahead, and potential methods to overcome these limits remain a mystery. Another consideration is that our ability to work with strangers is often influenced by stereotypes based on incomplete knowledge of an individual [4]. An investigation into the role of incomplete information about opponents in such a model deserves investigation - actors may still be capable of cooperation through assumption, but harmful stereotypes about another may lead to potentially worse results than no knowledge at all.

Chapter 8

Code Availability

The code for this project can be found at `https://github.com/urn216/StagHunt`. It requires Java 17 to run and may receive updates in the future.

Bibliography

[1] ALTMAN, A., BERCOVICI-BODEN, A., AND TENNENHOLTZ, M. Learning in one-shot strategic form games. In *Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings 17* (2006), Springer, pp. 6–17.

[2] AUMANN, R. Nash equilibria are not self-enforcing. *Economic decision making: Games, econometrics and optimisation* (1990), 201–206.

[3] BELLMAN, R. A markovian decision process. *Indiana Univ. Math. J. 6* (1957), 679–684.

[4] BURNETT, C., NORMAN, T. J., AND SYCARA, K. Bootstrapping trust evaluations through stereotypes. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)* (2010), International Foundation for Autonomous Agents and Multiagent Systems.

[5] DJIGUEMDE, M., DUBOIS, D., SAUQUET, A., AND TIDBALL, M. Continuous versus discrete time in dynamic common pool resource game experiments. *Environmental and Resource Economics 82*, 4 (2022), 985–1014.

[6] FISCHER, P., KRUEGER, J. I., GREITEMEYER, T., VOGRINCIC, C., KASTENMÜLLER, A., FREY, D., HEENE, M., WICHER, M., AND KAINBACHER, M. The bystander-effect: a meta-analytic review on bystander intervention in dangerous and non-dangerous emergencies. *Psychological bulletin 137*, 4 (2011), 517.

[7] FREAN, M., AND MARSLAND, S. Holds enable one-shot reciprocal exchange. *Proceedings of the Royal Society B 289*, 1980 (2022), 20220723.

[8] FUDENBERG, D., AND TIROLE, J. *Game theory*. MIT press, 1991.

[9] KOCHENDERFER, M. J., ET AL. *Algorithms For Decision Making*. MIT press, 2022.

[10] OSBORNE, M. J., ET AL. *An introduction to game theory*, vol. 3. Oxford University Press New York, 2004.

[11] OWEN, G. *Game theory*. Emerald Group Publishing, 2013.

[12] SKYRMS, B. *The Stag Hunt and The Evolution of Social Structure*. Cambridge University Press, 2004.