Multi-Agent Learning of Strategies in Abstract Argumentation Mechanisms

Rama Nemer

Master of Science in Information Technology
Faculty of Informatics
The British University in Dubai
2009

Abstract

Argumentation has been studied extensively in the field of Artificial Intelligence, however we know very little about its strategic aspects. This thesis aims to contribute to this general problem by examining the behavior of adaptive self-interested agents, in a multi-agent environment, over repeated encounters using game-theoretic techniques. I extended an existing simulation tool to implement argumentation games and used it to run repeated game experiments using combinations of characteristic argumentation games, adapted from literature, and types of adaptive agents under different conditions. The theme used was that of a court setting whereby there is a judge listening to arguments from different agents. Once all arguments have been presented, the judge must make a ruling: i.e decide which arguments are valid and hence which agents win by presenting them. Agents are assumed to be self-interested and adaptive so they may have conflicting preferences about which arguments they want the judge to accept and they can learn different strategies in order to achieve goals that reflect those preferences. The results indicate that the agents use a multitude of different strategies to influence the judge and maximize their payoff, thereby revealing different combinations of arguments with different frequencies, depending on the Nash equilibria of the game, the dominance of the pure strategies and the Pareto efficiency of the pure strategies in a game. These are dependent on aspects inherent in the argumentation game. While truth revelation was a dominant strategy in some games, interestingly in other cases the agents were able to gain a payoff that is higher than that of all the individual Nash equilibria by playing strategies involving combinations of the Nash equilibria. As for the effect of the learning algorithm on the choice of strategy, the results confirm that WPL is biased toward mixed strategies while GIGA is faster in convergence to pure strategy Nash equilibria. The importance of this kind of work lies in the fact that it combines two aspects of multi-agent systems that have been quite separate to-date: argumentation protocols and multi-agent learning in games.

Acknowledgment

I would like to express my sincere thanks and appreciation to my supervisors, Dr. Iyad Rahwan and Dr. Sherief Abdallah, for their inspiration, support and encouragement throughout the process of completing this thesis.

I am also grateful for the love and support provided by my parents and sister, along with other family members and friends who have all made an invaluable contribution each in his/her own special way.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Rama Nemer*)

Contents

Chapter 1

Overview

1.1 Introduction

In its most fundamental form, Argumentation is defined as a the act of using reason to control the acceptability of a standpoint by "putting forward a set of propositions (i.e. arguments) intended to support (or refute) the standpoint before a rational judge" [27]. Over a decade ago, Dung introduced the theory of abstract argumentation frameworks, to study the mechanisms humans use in argumentation, which placed the foundations of many studies into the implementation of argumentation mechanism on computers [9]. In this theory, arguments are represented on a very abstract level with binary defeat relations between them and there are rules for the acceptability of these arguments. Since then, argumentation systems have been further developed within research work in Artificial Intelligence [3], amongst other inter-related disciplines, that benefit the study and application of argumentation theory.

The most recent publication on Multi-Agent Systems loosely defines them as "systems that include multiple autonomous entities with either diverging information or diverging interests, or both" and attributes the capacity to learn in this context as "a key facet of intelligent behavior" [23]. This is to encompass the multitude of disciplines, besides computer science, over which the field spans, such as economics, philosophy and linguistics, and cover concepts such as logic, probability theory, game theory and optimization.

Imagine a court setting whereby there is a judge listening to arguments form different agents. Once all arguments have been presented, the judge must make a ruling: i.e decide which arguments are valid and hence which agents win by presenting them. The interesting part is that these agents are self-interested and adaptive so they may have conflicting preferences about which arguments they want the judge to accept and they can learn different strategies in order to achieve goals that reflect those preferences.

Within this scenario some interesting questions arise, the answers of which will contribute to the research in this field, such as how, and how well, agents learn argumentation strategies over repeated encounters with the judge and other agents.

This work combines two interesting aspects of multi-agent systems and hence raises new questions and potential areas of study. Specifically, learning of argumentation strategies amongst agents in a multi-agent environment, in terms of both theory and applications, which is supported by a growing interest within the field of Artificial Intelligence and amongst the Agents and Multi-Agents community.

1.2 Problem Statement

While argumentation has been studied extensively, we know very little about strategic aspects of argumentation. A substantial amount of work has been done on each of abstract argumentation and multi-agent learning of strategies, of which an overview is given in Chapter 2. This work is unique, in that it is an attempt at linking them together in an experimental study that allows quantitative experimentation, as well as shedding light on some interesting conclusions regarding their intersection.

Recently, Rahwan and Larson began addressing this issue by studying Dung's framework using game-theoretic techniques. They considered conditions under which agents have an incentive to tell the truth by revealing all their arguments. They presented the Argumentation Mechanism Design (ArgMD) [18] of which the details and significance are discussed in section 2.4.

However, there are situations in which these conditions do not hold an example being when multiple equilibria of strategies exist. Moreover, Rahwan and Larson considered single-shot games and did not study issues that arise when agents can be adaptive.

This thesis aims to contribute to the general problem of trying to understand strategic behavior in argumentation over repeated encounters when agents are adaptive.

1.3 Research Questions

This thesis addresses the following questions:

1-What kind of strategies the agents might use to influence the "judge", which is essentially the game itself, or the mechanism?

2-Will the agents learn over time to successfully manipulate the judge by lying or is the best strategy to always tell the truth?

3-Does the kind of adaptive agent, in terms of learning algorithm, have an effect on the strategies used by the agent in the same game? If so, what could that be attributed to?

1.4 Contribution

The contributions of this thesis to the general problem of trying to understand strategic behavior in argumentation over repeated encounters when agents are adaptive are (1) to provide a simulation tool to allow quantitative experimentation with a framework for implementing argumentation games and (2) to provide some insight by running argumentation games and analyze their results. This section outlines the answers to the research questions that were reached through these contributions.

1- *What kind of strategies the agents might use to influence the "judge", which is essentially the game itself, or the mechanism?*

To answer this question the argumentation game simulation tool was used to run a few games designed based on different scenarios by changing the parameters that define the argumentation game such as the argumentation graph, the assignment of arguments to the agents and distribution of utilities over the arguments. These parameters in turn define the game-theoretic aspects of the game and result in a multitude of different strategies that the agents will use depending on the Nash equilibria of the game, the dominance of the pure strategies strategies and the Pareto efficiency of the pure strategies. Intuitively this means revealing different combinations of arguments with different frequencies. In essence, the agents aim to maximize their pay so they either play the Nash equilibrium, either pure or mixed, or a combination of Nash equilibria that will achieve that goal. It is interesting to note that by playing a combination of

Nash equilibria, in some cases the agents were able to gain a pay that is higher than that of all the Nash equilibria.

2-*Will the agents learn over time to successfully manipulate the judge by lying or is the best strategy to always tell the truth?*

As mathematically proved by Rahwan and Larson [19], the results, across all games where the agents are not self defeating, are consistent with the conclusion that the game is indeed strategy proof and the dominant strategy equilibrium is to reveal all the arguments. From the agents perspective however, this is only one pay maximising strategy, amongst others, they might learn. With the exception of being under these conditions, the agents do learn strategies that involve lying, or hiding some of their arguments, in order to manipulate the judge. To answer the question of what motivates agents to use certain strategies over others, extensive experimentation beyond the scope of this thesis is required.

3-*Does the kind of adaptive agent, in terms of learning algorithm, have an effect on the strategies used by the agent in the same game? If so, what could that be attributed to?*

To determine whether different Multi-Agent Reinforcement Learning (MARL) algorithms converge to different equilibria in the same game, and the attributes of the algorithms that might cause this, the same games were ran using the two algorithms WPL and GIGA with different learning parameters. The results obtained were specific to the game, in some cases both algorithms converged to the same strategy under all circumstances while in others the same algorithm converged to different strategies when the learning parameters where changed. Generally however, WPL was found to be biased toward mixed strategies, even in games which had only pure strategy Nash equilibria, while GIGA was faster in convergence to pure strategy Nash equilibria.

1.5 Scope

The work in this thesis is an initial attempt at exploring strategic aspects of argumentation in games involving adaptive agents in a multi-agent system. The implemented tool and resulting games examined are only a basic subset of the infinite possibilities of scenarios that can be created. Essentially, in addition to the learning algorithms and variable learning parameters that were implemented in the existing tool, the grounded extension and a few parameters, reflecting the assignment of arguments to agents and utilities to winning these arguments, were added to define argumentation games.

Aside from the results achieved and contributions made, many more research questions arise that are beyond the scope of this thesis but worth investigating. Answering these questions will involve either partial mathematical proof, extensive experimentation and the extension of the simulation tool to implement more parameters that allow the representation of a wider variety of characteristics in games.

1.6 Organisation of Thesis

From this point on, this thesis is organised as follows: Chapter 2 provides some background on the relevant concepts used, in this study, in argumentation in general and in the field of Artificial Intelligence, abstract argumentation frameworks, game theory, argumentation games and multi-agent learning of strategies. Following that, Chapter 3 defines the specifications of the argumentation game and discusses the details of its implementation based on the existing tool. The experimental design and results of the five cases that were used as benchmark games are then discussed in Chapter 4. The conclusions of the thesis are finally stated in Chapter 5 along with a discussion of recommendations for further work. Appendices A, B and C give details of the technologies and tools used for implementation, an analysis and a sample run of the argumentation game and additional results that may be of interest, respectively.

Chapter 2

Background

2.1 Argumentation in Artificial Intelligence

Intuitively the term Argumentation is very broad and Argumentation Theory is an interdisciplinary field of research into which, amongst others, fields as diverse as philosophy and linguistics, alongside artificial intelligence have contributed to and benefited from. Many logic systems, ranging in levels of abstraction, with some similarities nonetheless, have been developed to formalize the reasoning behind "defeasible argumentation" which is the term referring to the process of logically evaluating the tenability of a claim by assessing arguments produced for and against it [17].

An excellent survey of the work achieved in the field of argumentation thus far with relevance to artificial intelligence is presented in a paper by Bench-Capon and Dunne [3]. The paper begins by examining the history of classical argumentation within philosophy through an overview of argumentation themes starting with the philosophical investigations by Aristotle to those by present day philosophers. A comparison is then made between argumentation and the traditional concepts of logical reasoning and mathematical proof, which are essentially acknowledged more by scientists in the field of artificial intelligence. The major contributions, such as Dung's extension based semantics as an argumentation model are discussed in significant detail and a presentation of the current trends in research concerning these topics is also made.

Recently a number of applications have emerged that rely on the use of argumentation amongst autonomous agents [21]. In some applications agents use argumentation amongst each other to persuade each other of certain actions, to negotiate and reach agreement regarding resource distribution [15]. Additionally, applications have been implemented, as an extension to expert systems, whereby agents formulate arguments to persuade humans to eat healthier food [14] or to give doctors advice based on medical expertise [10]. Research efforts are also being invested into defining argumentation schemes to allow for agent communication on the semantic grid [26] to allow for distributed argumentation amongst agents.

2.2 Abstract Argumentation Frameworks

An argument is a conclusion and a set of premises that can be used to derive it using some inference rules. A statement in the English language can therefore be represented as a sequence of sub-statements making up the premises and conclusion of an argument. It is defined formally as:

Definition 1 (Argument [4])**.** *An argument is a pair $\langle \Psi, \psi \rangle$ such that: (1) $\Psi \subseteq \Delta$; (2) $\Psi \nvdash \bot$ (3) $\Psi \vdash \psi$; and (4) there is no $\Psi\prime \subset \Psi$ such that $\Psi\prime \vdash \psi$. We say that $\langle \Psi, \psi \rangle$ is an argument for ψ. We call ψ the claim of the argument and Ψ the support of the argument (we also say that Ψ is a support for ψ).*

Dung's argumentation framework [9] is the most abstract of these argumentation systems and is described in detail in section 2.2 because this work heavily relies on it. Concepts from Prakken and Sartor's system were also used in this study to formulate the natural language (English) versions of the arguments, in the examples used in the experimentation, because it is inspired by legal reasoning and defines attack, or defeat, between arguments using the concepts of *rebutting* and *undercutting* [16]. These concepts, which provide a logical method for formulating arguments in the English language that are intuitive yet conform to the defeat [1] relations in an abstract argumentation graph, are defined intuitively as follows:

Definition 2 (Rebut). *An argument* $\langle \Psi, \psi \rangle$ *rebuts argument* $\langle \Phi, \phi \rangle$ *iff* $\psi \equiv \neg\phi$.

Hence, an argument A rebutts an argument B if and only if A *conclusion-to-conclusion* attacks B and rebutting is symmetric in the sense that if two arguments have contradicting conclusions then they mutually rebutt each other. This is used in cases where two arguments defeat each other.

Definition 3 (Undercut [12]). *An undercut for an argument* $\langle \Phi, \phi \rangle$ *is an argument* $\langle \Psi, \neg (\theta_1 \wedge \ldots \wedge \theta_n) \rangle$ *where* $\{\theta_1, \ldots, \theta_n\} \subseteq \Phi$.

An argument A undercuts an argument B if and only if A *conclusion-to-premises* attacks B. Undercutting is used to formulate arguments with a one way defeat relation using statements in English whereby the conclusion of one; the defeater, attacks the premises of the other; the defeated argument.

Definition 4 (Defeat [16]). *If one argument undercuts the other, and the other does not undercut but only rebutts the first, the first defeats the second but the second does not defeat the first.*

Therefore, it is not true that if A undercuts B, then B undercuts A and neither is it true that if A undercuts B, then B does not undercut A. In some cases if A rebuts B, then B rebuts A but it this does not always hold.

Since the publication of the paper titled "On the acceptability of arguments and its fundamental role in non monotonic reasoning, logic programming and n-person games" [9] over a decade ago, Dung's introduction of the theory of abstract argumentation frameworks has placed the foundations of many studies of implementing argumentation mechanism on computers. Essentially, such an argumentation framework is a logic-programming based model for reasoning about argumentation whereby arguments are represented on a very abstract level with binary defeat relations between them and there are rules for the acceptability of these arguments.

A brief outline of a few key concepts and definitions in Dung's argumentation system that are relevant to this study is given below beginning with the definition of an argumentation framework:[2]

Definition 5 (Argumentation framework [18]). *An* argumentation framework *is a pair* $AF = \langle \mathcal{A}, \rightharpoonup \rangle$ *where* $\mathcal{A}$ *is a set of arguments and* $\rightharpoonup \subseteq \mathcal{A} \times \mathcal{A}$ *is a defeat relation. We say that an argument* α defeats *an argument* β *iff* $(\alpha, \beta) \in \rightharpoonup$ *(sometimes written* $\alpha \rightharpoonup \beta$*).*

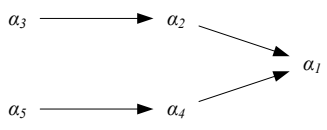


Figure 2.1: A simple argument graph

[1] Defeat and Attack are used interchangeably throughout this thesis.
[2] The definitions have been quoted from the referenced papers [18, 19, 9, 7].

Figure 2.1 above is an example of a representation of an argumentation framework as a graph whereby vertices are arguments and directed arcs are defeat relations among them. Argument α_1 has two defeaters α_2 and α_4, which are themselves defeated by arguments α_3 and α_5 respectively.

The notion of defense amongst arguments follows the intuition that an argument *defends* another argument if it defeats one of its defeaters. Thus in the above example, α_3 and α_5 both defend α_1. Furthermore, it can be implied that a set of arguments defends a given argument if it defeats all its defeaters: $\{\alpha_3, \alpha_5\}$ defends α_1. A set of arguments is defined to be *conflict free* if none of its arguments defeat each other.

As for the acceptability of arguments within a framework, the semantics are defined as follows [18]:

Definition 6 (Characteristic function)**.** *Let $AF = \langle \mathcal{A}, \rightharpoonup \rangle$ be an argumentation framework. The* characteristic function *of AF is $\mathcal{F}$: $2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ such that, given $S \subseteq \mathcal{A}$, we have $\mathcal{F}(S) = \{\alpha \in \mathcal{A} \mid S \text{ defends } \alpha\}$.*

Definition 7 (Acceptability semantics [18])**.** *Let S be a conflict-free set of arguments in framework $\langle \mathcal{A}, \rightharpoonup \rangle$.*

- *S is* admissible *iff it is conflict-free and defends any element in S (i.e. if $S \subseteq \mathcal{F}(S)$).*
- *S is a* complete extension *iff $S = \mathcal{F}(S)$.*
- *S is a* grounded extension *iff it is the minimal (w.r.t. set-inclusion) complete extension (or, alternatively, if S is the least fixed-point of $\mathcal{F}(.)$).*
- *S is a* preferred extension *iff it is a maximal (w.r.t. set-inclusion) complete extension (or, alternatively, if S is a maximal admissible set).*
- *S is a* stable extension *iff $S^+ = \mathcal{A} \backslash S$.*
- *S is a* semi-stable extension *iff S is a complete extension of which $S \cup S^+$ is maximal.*

Each of these semantics has the following intuition:

- A set of arguments is *admissible* if it is conflict-free set in which all the arguments are acceptable with respect to the set.
- If and only if a set contains all the arguments defended by it then it is a *complete extension*. There may be more than one complete extension.
- A *grounded extension* is a *'minimal' complete extension* and contains the arguments which are not defeated by any other arguments and the arguments that are defended by them. There exists only one grounded extension.
- A *preferred extension* includes the grounded extension but further maxi mises the accepted arguments as long as there is no inconsistency created by doing that.
- A set of arguments is a *stable extension* when it is a preferred extension that defeats each argument outside of it.
- A *semi-stable extension* is a preferred extension that maximizes the number of defeated arguments outside of it.

The determination of which arguments are accepted, rejected or the status of which can not be decided, within an argumentation framework, is carried out by *Argument Labeling* [7] defined as:

Definition 8 (Argument Labeling)**.** *Let $\langle \mathcal{A}, \rightharpoonup \rangle$ be an argumentation framework. An* argument labeling *is a total function $L : \mathcal{A} \rightarrow \{\texttt{in}, \texttt{out}, \texttt{undec}\}$ such that:*

Semantics	Restriction on Labeling	Extension-based Description
complete	all labelings	conflict-free fixpoint of $\mathcal{F}$
grounded	minimal `in` minimal `out` maximal `undec`	minimal fixpoint of $\mathcal{F}$ minimal complete extension
preferred	maximal `in` maximal `out`	maximal admissible set maximal complete extension
semi-stable	minimal `undec`	admissible set with max. $S \cup S^+$ complete ext. with max $S \cup S^+$
stable	empty `undec`	S defeating exactly $\mathcal{A} \backslash S$ conflict-free S defeating $\mathcal{A} \backslash S$ admissible set S defeating $\mathcal{A} \backslash S$ complete ext. S defeating $\mathcal{A} \backslash S$ preferred ext. S defeating $\mathcal{A} \backslash S$ semi-stable ext. S defeating $\mathcal{A} \backslash S$

Table 2.1: An overview of admissibility based semantics

- $\forall \alpha \in \mathcal{A} \ : \ (L(\alpha) = \texttt{out} \equiv \exists \beta \in \mathcal{A}$ *such that* $(\beta \rightharpoonup \alpha$ *and* $L(\beta) = \texttt{in}))$*; and*
- $\forall \alpha \in \mathcal{A} \ : \ (L(\alpha) = \texttt{in} \equiv \forall \beta \in \mathcal{A} \ : \ (\ $*if* $\beta \rightharpoonup \alpha$ *then* $L(\beta) = \texttt{out}))$

In other words, an argument is labeled in if and only if all its defeaters are labeled out and it is labeled out if and only if one of its defeaters is labeled in. The remaining arguments will be labeled undecided.

Table 2.1 summarizes the correspondence between argument labelings and extensions established by Caminada [7].

2.3 Game Theory

Game theory uses mathematics to provide tools for the analysis of the phenomena that occur when autonomous entities interact by providing a suitable set of concepts for analyzing strategic behavior and learning of strategies by self-interested agents[3]. Hence, it has been used in experimental studies on convergence in MARL [2] in addition to the mechanism design of the abstract argumentation mechanism for self-interested agents in a multi-agent environment [18], on which the experimental work in this thesis is based. Similarly, in the multi-agent environment presented in the context of this work, the same concepts are applied through modeling the agents' decisions using a repeated stochastic, from the perspective of one agent due to the uncertainty introduced by the other agents' actions [2], strategic/normal form game which captures the essence of interaction within a multi-agent system. These key concepts, taken from a more thorough field of Economics [13], will be introduced in this section along with correlation to their applicability within this context.

A game is a matrix that specifies the payoff that the players get in each of the different combinations of actions that they collectively play. For example, a normal form game with n players with the set of actions A_i, and the reward $R_i : A_1 \times ... \times A_n \rightarrow \Re$ based on the joint action of all the players, for each player i is represented by the tuple $\langle n, A_1, ..., A_n, R_1, ..., R_n \rangle$. A 2-player 2-action game can be represented as 2 by 2 payoff matrix whereby in each cell (i, j) is the payoffs player 1 (row) and player 2 (column) receive, respectively, if they play actions i and j respectively. Consequently, additional actions will be included as additional rows and games with 3 players or more can not be represented in a 2 dimensional matrix. Table 2.2 shows a general 2-player 2-action game and examples of classic game theory benchmark games.

[3]Agent and player are used interchangeably, to mean the same thing, throughout this thesis.

(a) 2-player-2-action game

	a1	a2
a1	r_{11}, c_{11}	r_{12}, c_{12}
a2	r_{21}, c_{21}	r_{22}, c_{22}

(b) coordination

	a1	a2
a1	2,1	0,0
a2	0,0	1,2

(c) matching pennies

	H	T
H	1,-1	-1,1
T	-1,1	1,-1

(d) tricky

	a1	a2
a1	0,3	3,2
a2	1,0	2,1

Table 2.2: Benchmark 2-player 2-action games

If I denotes a set of self-interested agents, then $\theta_i \in \Theta_i$ denotes the *type* of agent i, out of the set of possible types Θ_i, which defines the agent's preferences over *outcomes* $o \in \mathcal{O}$ in the set of all possible outcomes $\mathcal{O}$. The utility function $u_i(o, \theta_i)$ represents the agent's preferences whereby if $u_i(o_1, \theta_i) > u_i(o_2, \theta_i)$ then the agent prefers outcome o_1 over o_2. In strategic games amongst self-interested agents, the agents select actions that maximize their payoffs while competing against each other. This selection of actions is referred to as the agent's *strategy*, $s_i(\theta_i)$ for agent i, which is in essence a plan of what actions the agent will take each time it is required to act based on the information the agent has at that time. Hence, if Σ_i is the set of all possible strategies for agent i then $s_i(\theta_i) \in \Sigma_i$. When each agent i is playing strategy $s_i(\theta_i)$ the outcome is the *strategy profile*, also referred to as *joint policy*, $s = (s_1(\theta_1), \ldots, s_I(\theta_I))$. Note that $s_{-i}(\theta_{-i}) = (s_1(\theta_i), \ldots, s_{i-1}(\theta_{i-1}), s_{i+1}(\theta_{i+1}), \ldots, s_I(\theta_I))$ so $s = (s_i, s_{-i})$ and therefore $u_i((s_i, s_{-i}), \theta_i)$ is the utility of agent i, of type θ_i, when the agents play strategies specified by strategy profile $(s_i(\theta_i), s_{-i}(\theta_{-i}))$.

In game theory *solution concepts* are used to determine and examine the outcomes when rational strategic agents interact and in turn how self-interested agents choose their strategies to maximize their own utilities and collectively determine not only their own but other agents' outcomes as well. The *Nash equilibrium* is the most well known of these solution concepts and is defined intuitively as the strategy profile in which each agent is maximizing its own utility given the strategies of the other agents in the game [13], and formally in this context as:

Definition 9 (Nash Equilibrium [18])**.** *A strategy profile $s^* = (s_1^*, \ldots, s_I^*)$ is a Nash equilibrium if no agent has incentive to change its strategy, given that no other agent changes. Formally,*

$$\forall i, u_i(s_i^*, s_{-i}^*, \theta_i) \geq u_i(s_i', s_{-i}^*, \theta_i), \forall s_i'.$$

Hence, referring again to the games in table 2.2, the *coordination* game has two pure Nash equilibria: $[(0,1)_r, (0,1)_c]$ and $[(1,0)_r, (1,0)_c]$. Both the *matching-pennies* and the *tricky* games have one mixed Nash equilibrium where the actions are played with equal probability $[(\frac{1}{2}, \frac{1}{2})_r, (\frac{1}{2}, \frac{1}{2})_c]$. Since a game can have multiple, and combinations of pure and mixed strategy, Nash equilibria agents will be uncertain toward which one to play and that is assuming they are able to determine them since that requires the agents to know all other agent's preferences. A solution concept that does not have this constraint is the *dominant-strategy equilibrium* which is a strategy profile in which each agent is playing its *dominant* strategy [13], which is the strategy that maximizes an agent's utility regardless of the strategies played by the other agents and in this context is formally defined as:

Definition 10 (Dominant Strategy [18])**.** *Strategy s_i^* is dominant if*

$$u_i(s_i^*, s_{-i}, \theta_i) \geq u_i(s_i', s_{-i}, \theta_i)\ \forall s_{-i},\ \forall s_i'$$

This is the weak form of dominance whereby the dominant strategy results in a greater utility in at least one action and equal utility in the rest. A more strict notion of dominance is when $u_i(s_i^*, s_{-i}, \theta_i) > u_i(s_i', s_{-i}, \theta_i)\ \forall s_{-i},\ \forall s_i'$ in which case s_i^* is said to strictly dominate s_i' since it results in a greater utility for the agent in every case. A dominant strategy is therefore a Nash equilibrium but not necessarily the opposite. Additionally, there are strategic games in which no agent has a dominant strategy.

Another solution concept which is used to analyze the results of this experimentation is that of Pareto dominance [13] (and optimality) defined formally in this context as:

Definition 11 (Pareto Dominance [19]). *An outcome* $o_1 \in \mathcal{O}$ Pareto dominates *outcome* $o_2 \neq o_1$ *iff* $\forall i \in I$, $o_2 \succeq_i o_1$ *and* $\exists j \in I$, $o_2 \succ_j o_1$.

Definition 12 (Pareto Optimality [19]). *An outcome* $o_1 \in \mathcal{O}$ *is* Pareto optimal *(or* Pareto efficient*) if there is no other outcome* $o_2 \neq o_1$ *such that* $\forall i \in I$, $o_2 \succeq_i o_1$ *and* $\exists j \in I$, $o_2 \succ_j o_1$.

An outcome is Pareto dominated, by another outcome, if the agent will make an improvement by choosing another outcome over it without making any of the other agents worse off. Therefore, an outcome is Pareto optimal if it is not Pareto dominated by any other outcome.

2.4 Argumentation Games

Mechanism design is a framework initially developed as part of economic theory to addresses the problem of ensuring that a desirable system-wide outcome is reached as a result of the interaction of several parties that have different preferences over the outcome. This section explains an Argumentation Game that is developed based on this framework.

Rahwan and Larson's research introduces argumentation mechanism design (ArgMD) to fill the niche for a method for the purpose of "understanding the strategic aspects of abstract argumentation among self-interested agents" [18]. This takes into account not only the rules of argument acceptability or labeling but also covers the possible scenario whereby agents follow strategies when deciding what arguments to present or hide, thereby 'lying', while competing amongst other agents which have conflicting interests over the arguments they want to be accepted. The work relies on a few concepts from game theory and mechanism design that are applied in the context of multi-agent argumentation to define an Argumentation game.

In such an environment, the type of an agent represents the private information (known only by the agent) and the preferences of the agent over a possible set of outcomes. These preferences are expressed by a utility function that depends on the agent type and outcome to give utility values to each outcome.

Definition 13 (Agent Type [18]). *Given an argumentation framework* $\langle \mathcal{A}, \rightharpoonup \rangle$, *the* type *of agent* i, $\mathcal{A}_i \subseteq \mathcal{A}$, *is the set of arguments that the agent is capable of putting forward.*

An agent's strategy is the plan that the agent follows to determine its actions at each time it is required to act during a game. Since the agents are self-interested, their goals will be to maximize their own utility, and they will choose strategies accordingly.

Thus an Argumentation game whereby a group of self-interested agents interact, in the context of this work, is defined as a mechanism whereby each of the agents involved will simultaneously decide on the subset of the set of their arguments that they wish to reveal, hence hiding the rest, and then the mechanism will label the accepted arguments based on the argument acceptability criterion used. This is further extended based on the sceptical evaluation mechanism [18] to define the skeptical direct argumentation game which will calculate the grounded extension given all the arguments presented by all the agents.

Definition 14 (Sceptical Argumentation Game). *A* sceptical argumentation game *for argumentation framework* $\langle \mathcal{A}, \rightharpoonup \rangle$ *is* $(\Sigma_1, \ldots, \Sigma_I, g(.))$ *where:*

- $\Sigma_i \in 2^{\mathcal{A}}$ *is the set of strategies available to each agent;*
- $g : \Sigma_1 \times \cdots \times \Sigma_I \rightarrow 2^{\mathcal{A}}$ *is an outcome rule defined as:* $g(\mathcal{A}_1^\circ, \ldots, \mathcal{A}_I^\circ) = Acc(\langle \mathcal{A}_1^\circ \cup \cdots \cup \mathcal{A}_I^\circ, \rightharpoonup \rangle, \mathcal{S}^{grnd})$ *where* $\mathcal{S}^{grnd}$ *denotes sceptical acceptability semantics.*

Rahwan and Larson prove this mechanism to be strategy-proof under the condition that the agents don't have arguments that directly/indirectly attack each other and when all the agents have acceptability maximizing preferences (their aim is to get the maximum number of their arguments accepted) [19]. If $\mathcal{LAB}$ is the set of all possible legal labelings of all arguments put forward by participating agents, then acceptability maximising preferences is defined as follows:

Definition 15 (Acceptability maximising preferences [19]). *An agent i has* individual acceptability maximising preferences *iff* $\forall L_1, L_2 \in \mathcal{LAB}$ *such that* $|\mathtt{in}(L_1) \cap \mathcal{A}_i| \geq |\mathtt{in}(L_2) \cap \mathcal{A}_i|$, *we have* $L_1 \succeq_i L_2$.

In other words, even though agents have an incentive to lie, by hiding arguments in order to maximize their utility, under this mechanism, and the condition that each agents' arguments do not self defeat, the dominant strategy is for an agent to reveal all of its arguments.

2.5 Multi-Agent Learning of Strategies

In the case of single-agent learning only one agent is interacting, and learning from, a dynamic and unknown environment. A Markov Decision Process [11], consisting of a set of states, a set of actions, a transition function and a reward function, is an adequate representation of the single agent situation and a rational reinforcement learning algorithm such as Q-Learning [28] can find the required optimal policies in MDPs.

In a multi-agent system however, due to the presence of other adaptive agents, from the perspective of any one agent, the best action to be taken at any time will depend on the actions and changing policies of the other agents that are unknown [23]. Game theory, using concepts explained in detail in section 2.3, is an appropriate model and has been used extensively to test the performance of Multi-Agent Reinforcement Learning (MARL) algorithms by modeling them as repeated stochastic strategic/normal form games.

Bowling and Veloso defined *Rationality* and *Convergence* as required properties of a MARL algorithm and used them to rate the performance of different algorithms in a spectrum of stochastic games.

Definition 16 (Rationality [6]). *If the other players' policies converge to stationary policies then the learning algorithm will converge to a policy that is a best response to the other players' policies.*

Definition 17 (Convergence [6]). *The learner will necessarily converge to a stationary policy. This property will usually be conditioned on the other agents using an algorithm from some class of learning algorithms.*

Greedy learners [8], the collective name for Independent Learners (IL), which ignore other agents and use Q-learning, and Joint Action Learners (JAL), which estimate other agent's policy using fictitious play, have not performed as well in multi-agent environments as they do for single-agents [6]. Regardless of the fact that these algorithms only learn pure Nash equilibria and their convergence in competitive games has not been formally proved, they require that the agents have, or have the ability to acquire, some background knowledge of the other agents and the structure of the game itself. This is not the case in more realistic applications where the world is not visible to the agents and the domain is competitive [2].

Based on this need for an algorithm that learns a stochastic policy in iterated games in an open domain, Singh, Kears and Mansour introduced the Gradient Ascent algorithm whereby a player will move toward a strategy following the gradient with a step size δ [24]. The study proved that the algorithm is rational but not convergent. They further examined the algorithm using an infinitesimal step size, to see the effects on convergence as δ approaches 0 and called the refined algorithm Infinitesimal Gradient Ascent (IGA) which was further generalized to produce the Generalized IGA (GIGA) algorithm [29]. The observed convergence of either does not suffice since both of them did not consistently converge to a Nash Equilibrium especially in games with only mixed Nash Equilibria [2].

Hence, to improve convergence, Bowling and Veloso introduced the concept of a variable learning in IGA, whereby the size of the steps taken toward the policy varies, and a new learning technique named the "Win or Learn Fast" (WoLF) principle, , which uses a variable learning rate [6]. In WoLF the size of the step is dependent on whether the player is winning or losing.

It increases as the player in losing and decreases as the player in winning which translates intuitively into the notion that when the player is doing well and winning half the job is done since it's on the right track so it can slow down while if it is losing it needs to change it's direction, or strategy, faster.

Abdallah and Lesser examined the same issue of convergence through experimentation with different learning algorithms resulting in the introduction of the "Weighted Policy Learner" (WPL) which is gradient ascent learning algorithm that they show to converge in a variety of games, including the challenging Shapley's game [2]. The algorithm, when implemented in 2-action games, slows down learning if agents are moving towards a pure policy thus damping policy oscillation until the joint policy gradually stabilizes to the Nash quilibrium.

In this analysis, attention was limited to GIGA and WPL as a sample subset of MARL algorithms. Both of the algorithms share the four parameters below:

- The policy of an agent i denoted by π_i.
- The value-learning-rate α is used to compute the expected reward of an action a at time t, or $r^t(a)$ using the equation $r^{t+1}(a) \leftarrow \alpha R^t + (1-\alpha)r^t(a)$, where R^t is the sample reward received at time t and $0 \leq \alpha \leq 1$ [25].
- The policy-learning-rate δ ($\delta \to 0$) is the size of the step used to update the policy π.
- The exploration rate ϵ ($\epsilon \to 0$) is the minimum rate at which the algorithm deviates to a random action to explore.

The details of the way these two algorithms work differently are discussed in the following two sections.

2.5.1 Generalized Infinitesimal Gradient Ascent (GIGA) [29]

GIGA is a gradient ascent algorithm an agent i's policy π_i is updated based on the the value function $r_i(t)$, which is the gradient of expected payoffs, as in the following equations:

$$\Delta\pi_i^{t+1} \leftarrow \delta \frac{\partial r_i(t)}{\partial \pi_i}$$
$$\pi_i^{t+1} \leftarrow projection(\pi_i^t + \Delta\pi_i^{t+1})$$

The *generalization* of IGA that GIGA introduces is through defining the *projection* function as $projection(x) = argmin_{x':valid(x')}|x - x'|$, where $|x - x'|$ is the Euclidean distance between x and x'. The projection function projects an invalid policy to the *closest* valid policy within the unit simplex [2] that fulfills the constraints $\forall a \in A : 1 \geq \pi(a) \geq 0$ and $\sum_\pi = \sum_{a \in A} \pi(a) = 1$.

2.5.2 Weighted Policy Learner (WPL) [1]

WPL uses the policy gradient Δ to update policy π_i, and a *projection* function adopted from that of GIGA with the modification $\forall a : 1 \geq \pi(a) \geq \epsilon$, as shown in the following update equations:

$$\Delta\pi_i(a) \leftarrow \frac{\partial V_i(\pi)}{\partial \pi_i(a)} \cdot \delta \cdot \begin{cases} \pi_i(a) & \text{if } \frac{\partial V_i(\pi)}{\partial \pi_i(a)} < 0 \\ 1 - \pi_i(a) & \text{otherwise} \end{cases}$$
$$\pi_i \leftarrow projection(\pi_i + \Delta\pi_i)$$

Intuitively, this means that, the probability of choosing an action increases, if it is a good choice, or decreases, if it is a bad choice, at a rate that decreases as the algorithm moves toward

or away from it respectively. So the algorithm slows down gradually as long as it is moving toward an action but starts to learn fast as soon as it starts moving away from it.

Chapter 3

Specification and Implementation

3.1 Basic Simulator Architecture

The Java-based simulation tool is an extension of the tool implemented by Abdallah and Lesser in order to experiment with the convergence properties of the "Weighted Policy Learner" (WPL) [1], which is discussed in detail in section 2.5.

3.1.1 Existing Tool

The main requirement that the specification of the existing tool calls for is the provision of the possibility to examine the way the policies of multiple learning agents evolve over time as the agents learn, as opposed to just showing the final result. Hence, the results can be analyzed to reveal how the learning algorithm moves toward convergence.

In terms of games, this tool implements the classic games that are commonly studied in Game Theory. Namely, matching pennies, the coordination game and the tricky game, which are 2-player-2-action games, in addition to rock-paper-scissors and Shapley's, which are 2-player-3-action games and Jordan's, which is a 3-player-2-action game. As for learning algorithms, the tool implements players that will learn using either the Generalized Infitismal Gradient Ascent (GIGA), GIGA-Win or Learn Fast (GIGA-WoLF) [5] or the Weighted Policy Learner (WPL) learning algorithm. The number of steps in a game (number of encounters) and the initial policy π of the players are variable and can be set as parameters when running a game in addition to several parameters specific to the algorithm such as the value learning rate α, the policy learning rate δ and the exploration rate ϵ.

The tool comprises of the controller class which initializes the game, sets up the log, creates the game and the players, based on all the input parameters and, in essence, runs the game by pulsing through the time steps. In each step, the actions of each player are gathered and the payoff of each player is determined from the game's payoff matrix. The games, in their payoff matrix representation, are implemented in the game class, that extends a game interface, and the different types players, as learning algorithms, are implemented in a separate class each. The action, pay and updated policy of each player at each step is logged in a log file for later graphing and analysis.

3.1.2 Extension of Existing Tool

The major part of the extension of the tool was the realization of the Argumentation game. This was done as an extension of the game interface by implementing the class GameArgue along with a few peripheral classes. The argumentation game takes into account the provision of the capability of the representation of the concepts of abstract argumentation frameworks

and argumentation mechanism design as discussed in sections 2.2 and 2.4 respectively.

For instance, the representation of the argumentation framework as a collection of arguments with defeat relations between them is applied using an object, an instance of the class ArgumentGraph, that is a set of objects, instances of the class Argument, which have vectors of other Arguments as defeaters and victims (instance variables of an Argument object). Each argument also has an instance variable to hold its status once the graph is analyzed and its arguments labeled according the grounded extension.

The agent type, defined as the set of arguments that the agent is capable to put forward, or reveal, is determined as an input parameter and maintained as a variable in the game class. From it the set of all possible combinations of the agent's assigned arguments, the power set of the set of arguments, is calculated and also reserved as the agent's set of actions.

For each agent, the utility of winning each argument, or in other words getting that argument accepted, is also an input parameter. This allows the manipulation of the agents' incentives and preferences. A higher utility allocated to an argument, for an agent, will motivate the rational agent to essentially value it's acceptance more than another argument with lower utility. A negative utility would, in turn, prompt an agent to adopt a strategy that inhibits the said argument's acceptance. Hence, the concept of acceptance maximizing preference would correspond to all arguments having positive utilities.

Algorithm 3.1 describes how the game uses the input parameters to simulate the interaction of the agents and produce output logs. It is apparent from the cases discussed in Chapter 4 that each of the argumentation games can be represented as a generic matrix game and hence, instead of using a configuration file to initialize the each game and go through this algorithm, it could have been manually computed and hardcoded into the tool. While this may seem more efficient in the simple cases involving a few agents and a few arguments, the argumentation game tool was designed to account for further extension for use in more complex cases involving a larger number of agents and more complicated argumentation graphs. In that case this algorithm, whereby the sub-graph, extension and pay are calculated at each step, is definitely more appropriate.

Algorithm 3.1: The Argumentation Game

Input: *ArgGameParams.txt*: configuration file (arguments, attacks, agents and utilities)
log: Log file-base
p: Player(Learning Algorithm)
s: No of Steps
π: Initial policy
α: Learning rate
δ: Policy learning rate
ϵ: Exploration rate
Output: Log of policy and pay per agent per step
Create Argumentation Game(*log*,*s*)
Get Arguementation Game Parameters("ArgGameParams.txt");
Create Argumentation Graph(arguments, attacks);
foreach *Agent a* **do**
  Create Player(p, π, α, δ, ϵ);
  Create Actions(agents, arguments, utilities);
end
foreach *Step s* **do**
  foreach *Agent a* **do**
    Get Action;
  end
  Create ActionGraph;
  Compute Extenstion(ActionGraph);
  foreach *Agent a* **do**
    Calculate Pay;
    Update Policy;
  end
  Log Policy and Pay (*log*, *s*);
end

Dung's Grounded Extension

As defined in section 2.2, the grounded extension includes arguments which neither have any direct defeaters nor are defeated by any accepted arguments. Algorithm 3.2, used to label the arguments in an argumentation graph based on the grounded extension, is adapted from Dung's conclusion [9] that the grounded extension can be obtained by applying the characteristic function to the empty set iteratively [18].

Algorithm 3.2: The Grounded Extension of an Argumentation Graph

```
Input: Argumentation Graph
Output: Labeling of Argument Nodes to in, out and undec according to Grounded
        Extension
while Change = True do
    Change = False;
    foreach Argument α_i do
        Get Defeaters(α_i);
        if Defeaters(α_i) ≡ 0 then
            Change = True;
            Status(α_i) = in;
            foreach Argument α_j defeated by α_i do
                Status(α_j) = out;
                Remove α_j from the defeaters list of its own victims;
                Remove α_j from the graph;
            end
            Remove α_i from the defeaters list of its own victims;
            Remove α_i from the graph;
        end
    end
end
foreach Remaining Argument α_k do
    Status(α_k) = undec;
    Remove α_k from the graph;
end
```

3.1.3 Game Parameters and Output

The purpose and use of each of the parameters required and employed in the argumentation game is discussed here. The game utilizes the variable input parameters in the original multi agent learning tool in addition to a few other parameters that are specific to the argumentation and argumentation mechanism design aspect of the game. Refer to Appendix B for the details of how these parameters are formatted as input into the game and the resulting format and method of interpretation of the output.

General Game Parameters

The following are the general game parameters that are required to run the general multi agent learning tool.

- Game: Essentially, the name of the game to be played. This would be either "argue" for the argumentation game or a different corresponding name for one of the other supported games.
- Player: The player (learning algorithm); WPL, GIGA or GIGA-WoLF.
- Steps: The number of times (encounters) that the game is played.
- π: The initial policy of the players.
- α: The value learning rate of the learning algorithm which is the size of the step used to update the action values.
- δ: The policy learning rate of the learning algorithm which is the size of the step used to update the policy.

- ϵ: The exploration rate which is the minimum rate at which the algorithm deviates to a random action to explore.

Argumentation Game Parameters

The following are the Argumentation Game specific parameters which are read from an input configuration file.

- The arguments in the game.
- The argumentation framework of the game.
- The agents in the game.
- The agent's assigned arguments and their associated utilities.

3.2 Technical Limitations

The tool is Java-based and Eclipse SDK, an open source integrated development environment (IDE), was used to develop the extension also in Java. Matlab was used for the plotting, and further analysis, of the output data into results and Gambit was used for the game-theoretic analysis of the games. Refer to Appendix A for the details of, and justification for, the technologies used.

The technical limitations of the argumentation game tool for the most part be resolved given more time since they only require the extension of the simulation tool by implementing more parameters. Had these been implemented many characteristics of agents and judges could be modeled resulting in a wider variety of interesting scenarios worth studying as mentioned in chapter 5 as suggestions for further work.

Two particular limitations that impacted the work in this thesis are; Firstly the fact that utilities are assigned to the acceptability of individual arguments and not combinations of arguments. This forced the representation of the "Argumentative Battle of the Sexes" game using the general game tool, as a matrix game, for an accurate representation of the true intuition of the game. Secondly, a more intuitive user interface would have immensely helped by making the parameterization of games easier and decreased the chance of error and the need for double checking.

Chapter 4

Experimental Design and Results

4.1 Introduction

This chapter presents five cases of argumentation games each with different interesting characteristics and discusses the results achieved when these games are run using the implemented simulation tool under different conditions.

4.2 Parameter Design

In addition to using characteristic argumentation graphs from literature, that highlight different properties, to experiments with different argumentation scenarios, parameters such as the utilities assigned to the arguments, the assignment of the arguments to the agents, the learning algorithms used and their inherent learning parameters were all used as parameters to design the cases.

To ensure that the analysis is not tied to a particular algorithm two learning algorithms, WPL and GIGA, which have been discussed in detail in section 2.5 were used to evaluate the argumentation games.

4.3 Benchmark Games

4.3.1 Case 1: The "Nixon Diamond" Argumentation Game

The argument graph shown in figure 4.1 represents the argumentation framework of a famous and commonly studied scenario in non monotonic reasoning known as the "Nixon Diamond". In natural language α_1 represents the argument "Richard Nixon is anti-pacifist since he is a republican", and α_2 represents the argument "Richard Nixon is a pacifist since he is a quaker". Based on the knowledge that quakers are pacifist, republicans are not pacifist and Richard Nixon is both a quaker and a republican, the arguments attack each other. This framework, along with a natural language assignment to the arguments is used as an example to demonstrate the mutually inconsistent conclusions that result from default assumptions and thereby links the difference between the two preferred extension, in which one of the arguments is accepted each (credulous semantics), versus the grounded extension which is empty and rejects both arguments (skeptical semantics), using the complete extension [9].

Table 4.1 shows the argument assignments of the two agents in the "Nixon Diamond" and table 4.2 is its representation as a two player argumentation game. Each agent is given one argument, agent 1 given argument α_1 and agent 2 given argument α_2, with an acceptance utility

of 1 each. For each agent the action *Hide* is to hide its argument, in other words reveal the empty set {}, and *Show* is to reveal the assigned argument $\{\alpha_1\}$ for agent 1 and $\{\alpha_2\}$ for agent 2.

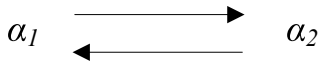


Figure 4.1: Game 1 - Arg Graph

Player	Argument (Utility)
p0	α_1 (1)
p1	α_2 (1)

Table 4.1: Game 1

	Hide	Show
Hide	0 0	0 1
Show	1 0	0 0

Table 4.2: Game 1 Representation

For both agents the action *Show* weakly dominates the action *Hide* and the game has three pure Nash equilibria:

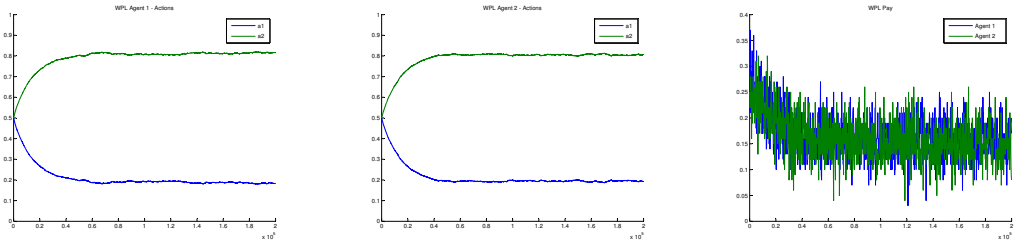
- $[(0,1)_{p1},(1,0)_{p2}]$
- $[(1,0)_{p1},(0,1)_{p2}]$
- $[(0,1)_{p1},(0,1)_{p2}]$

Two of the Nash equilibria are when one agent chooses to *Show* and the other to *Hide*, and vice versa, and the third in which both will choose to *Show*. This third Nash equilibrium is not Pareto efficient/optimal because by moving to one of the other two one of the agents will gain more utility without decreasing the utility of the other, in other words it does not achieve maximum welfare. However, since the agents are self interested and utility maximising, each only cares about its own utility and not maximum welfare, they do not favor the first two Nash equilibria over the third because a Pareto improvement might have been a motivation only as a result of collaboration, which is not the case here.

As seen in the graphs below, both WPL and GIGA converge to the pure Nash equilibrium $[(0,1)_{p1},(0,1)_{p2}]$ but under different parameters, specifically different values of the learning rate α. It can also be stated that GIGA converges better and faster than WPL and this is because WPL is biased against pure Nash equilibria and prefers mixed strategies.

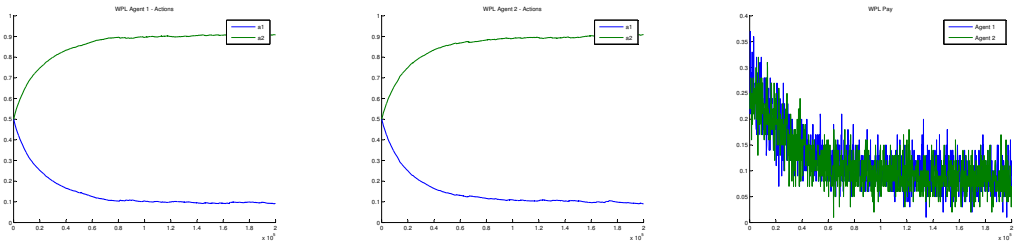
Figures 4.2, 4.3 and 4.4 show how the convergence of WPL gets closer to the pure Nash equilibrium $[(0,1)_{p1},(0,1)_{p2}]$ as α decreases. When α is 0.1 it converges to a mixed strategy of

$[(\frac{1}{5}, \frac{4}{5})_{p1}, (\frac{1}{5}, \frac{4}{5})_{p2}]$ and the pay of both agents converges to a value of roughly (0.15,0.15), then when α is 0.01 it converges to $[(\frac{1}{10}, \frac{9}{10})_{p1}, (\frac{1}{10}, \frac{9}{10})_{p2}]$ and the pay to (0.15,0.15) which is closer to the pure Nash equilibrium $[(0,1)_{p1}, (0,1)_{p2}]$. When α is further decreased to 0.001 it gets even closer by converging to $[(\frac{1}{20}, \frac{19}{20})_{p1}, (\frac{1}{20}, \frac{19}{20})_{p2}]$ with the pay converging to (0.05,0.05). It is interesting to note here that for the Nash equilibrium, the payoff is actually (0,0), which is lower than the equilibrium reached by WPL.



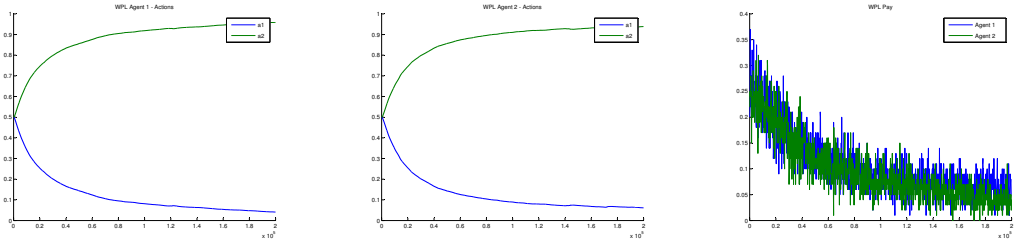
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.2: Game 1: WPL α=0.1 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

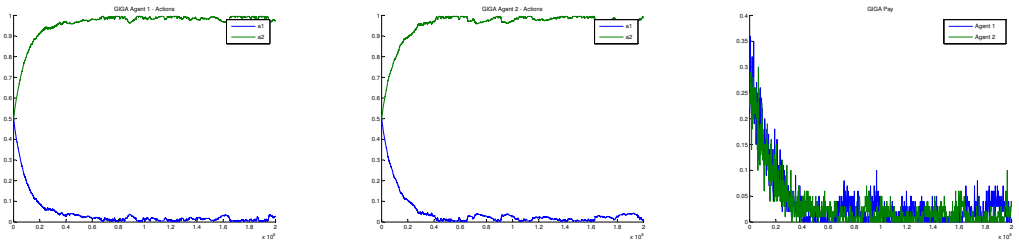
Figure 4.3: Game 1: WPL α=0.01 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

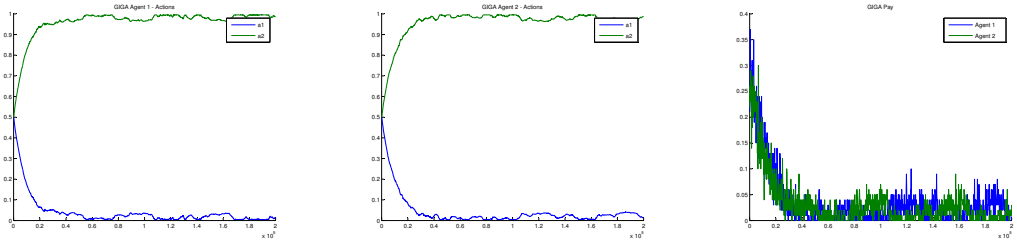
Figure 4.4: Game 1: WPL α=0.001 δ=0.002 ϵ=0.01

GIGA, on the other hand, performs better when α is larger and worse as it decreases. As seen in figures 4.5, 4.6 and 4.7 the agents' policies converge to the pure Nash Equilibrium $[(0,1)_{p1},(0,1)_{p2}]$ accurately enough when α is 0.1 and 0.01 along with the pay converging to almost (0,0). However when α is decreased to 0.001 at some point the policies start to deviate away from this Nash equilibrium for some time before they return, creating dips in the policy graphs and oscillations in the pay graphs. This behavior is due to the fact the policy should actually follow the value, so the value learning rate α should be lower than the policy learning rate δ.



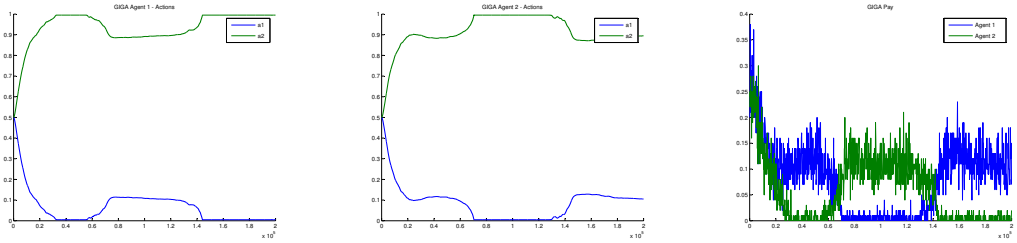
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.5: Game 1: GIGA α=0.1 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.6: Game 1: GIGA α=0.01 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.7: Game 1: GIGA α=0.001 δ=0.002 ϵ=0.01

4.3.2 Case 2: Contradicting Arguments

John is being prosecuted as one of the suspects in a bank robbery that occurred last week based on the prosecutor's argument that the characteristics of the robbery indicate that John is a prime suspect. The prosecutor also has proof that John has been in jail for the past month serving a sentence for a previous similar bank robbery. This does not directly support his argument that John is guilty, it actually contradicts it and thus attacks it directly, but it is proof of his previous involvement in such crimes and hence indirectly attacks John's lawyer's argument that he has a clear criminal record. In addition to claiming that John has a clean criminal record, which the lawyer uses to defeat the prosecutor's claim that John is the prime suspect, the lawyer can also make the argument that John indeed has been previously charged with robbery but serving his sentence in jail has made him learn his lesson.

Using the following statements:
q: "John has been in jail for the past month"
p: "John was in jail the day of the robbery"

r: "John has a clean criminal record"
s: "John is a prime suspect of the robbery"
v: "John is deterred by jail after serving previous sentence"
t: "John is a very honorable man in the community"
u: "There is evidence of John's involvement in the robbery"

The following explicit arguments were constructed then further summarized into shorter arguments that carry their implicit meaning.
α_1: $q, q \rightarrow p \vdash p$
"John has been in jail for the past month, if he was in jail for the past month then he was in jail on the day of the robbery. Hence he was in jail on the day of the robbery"
α_2: $(\neg p \wedge t), (\neg p \wedge t) \rightarrow (r \wedge \neg s) \vdash (r \wedge \neg s)$
"John was not in jail on the day of the robbery and is a very honorable man, people who are not in jail and are known to be honorable have clean criminal records and are not prime suspects of crimes. Hence he has a clean criminal record and he is not the prime suspect of the robbery"
α_3: $(s \wedge u), (s \wedge u) \rightarrow \neg p \vdash \neg p$
"John is the prime suspect of last week's bank robbery and there is evidence of his involvement in it, if someone is a prime suspect in a crime and there is evidence of their involvement in it then they were not in jail at the time it occurred. Hence he was not in jail on the day of the robbery"
α_4: $v, v \rightarrow \neg r \vdash \neg r$
"John is deterred by jail (he has learned his lesson after his last sentence), if someone is deterred by jail then they have been jailed before and so they don't have a clean criminal record. Hence he does not have a clean criminal record"

Figure 4.8 shows the argumentation graph corresponding to this scenario and these are simply shorter versions of the above detailed arguments:
α_1: "John has been in jail for the past month and hence was in jail on the day of the robbery"
α_2: "John was not in jail on the day of the robbery and is a very honorable man, hence he has a clean criminal record and he is not the prime suspect of the robbery"
α_3: "John is the prime suspect of last week's bank robbery and there is evidence of his involvement in it hence not in jail on the day of the robbery"
α_4: "John is deterred by jail, he has learned his lesson after his last sentence, hence he does not have a clean criminal record"

Agent 1, representing the lawyer, is given arguments α_2 and α_4, which attack each other, and agent 2, the prosecutor, is given arguments α_1 and α_3, which also attack each other. Additionally argument α_1 attacks α_2 which itself attacks α_3. Table 4.4 shows the agent types and the utilities of winning each of the arguments. A higher utility of 2 is given to α_2 than 1 to α_4 because assuming that the lawyer's preferred outcome is that John's reputation is reclaimed then winning the argument that he has a clean criminal record would be ideal. However, because the prosecutor might use his argument to attack it, the argument that John has learned his lesson is a backup argument that is weaker in terms of proving John's innocence but more is more likely to be won because none of the prosecutor's arguments attack it. So the lawyer has to weigh out whether to reveal that John does have a past criminal record, but has learned his lesson after his last sentence, or to stay quiet about that and directly use the argument that John has a clean criminal record and running the risk of it being attacked. Likewise, the prosecutor is more interested in winning the argument that John is the prime suspect, and only using the proof that he's been in jail to reinforce it, higher utility of 3 is given to α_3 than 1 to α_1.

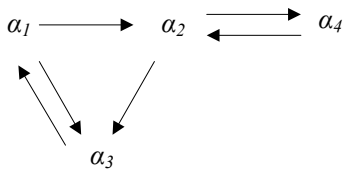


Figure 4.8: Game 2 - Arg Graph

Player	Argument (Utility)
p0	α_2 (2), α_4(1)
p1	α_1 (1), α_3(3)

Table 4.3: Game 2

	{}	$\{\alpha_3\}$	$\{\alpha_1\}$	$\{\alpha_1,\alpha_3\}$
{}	0 0	0 2	0 1	0 0
$\{\alpha_4\}$	1 0	1 2	1 1	1 0
$\{\alpha_2\}$	2 0	2 0	0 1	0 0
$\{\alpha_2,\alpha_4\}$	0 0	0 0	1 1	0 0

Table 4.4: Game 2 Representation

As can be deduced from table 4.4, for agent 1, the lawyer, (1,0,0,0) is strictly dominated by (0,1,0,0), which means that he is always better off revealing only α_4 than if he stays quiet and does not say anything. In addition (0,0,0,1) is weakly dominated by (0,1,0,0) which means that in some cases he is better off revealing only α_4 and not α_2 rather than revealing both. For player 2, the prosecutor, (1,0,0,0) and (0,0,0,1) are strictly dominated by (0,0,1,0) and weakly dominated by (0,1,0,0) which means that he is always better off revealing α_1 than keeping quiet or revealing both his arguments, and in all cases he is either the same or better off revealing only α_3 than keeping quiet or revealing both his arguments.

This leads to an interesting combination of Nash equilibria whereby one is pure, one is a mixed strategy and one is a combination whereby agent 1 has a mixed strategy while agent 2 has a pure strategy. The pure Nash Equilibrium $[(0,0,0,1)_{p1},(0,0,1,0)_{p2}]$, however, is not a dominant strategy equilibrium because for agent 1 (0,0,0,1), revealing both arguments, is a weakly dominated strategy.

- $[(0,0,0,1)_{p1},(0,0,1,0)_{p2}]$
- $[(0,\frac{1}{2},\frac{1}{2},0)_{p1},(0,\frac{1}{2},\frac{1}{2},0)_{p2}]$
- $[(0,\frac{1}{2},0,\frac{1}{2})_{p1},(0,0,1,0)_{p2}]$

Intuitively, it is apparent that a good plan for each would be to mix between revealing each of their two arguments equally and this translates to the mixed strategy Nash equilibrium. The prosecutor prefers to win α_3, which is the more powerful argument that is assigned higher utility, but needs to defend it using α_1 and so has the option of giving it up and just settling for α_1 which is weaker and had less utility but is better than zero utility. The lawyer prefers to win α_2 but has to settle for α_4 because α_2 is defeated by the prosecutor's argument α_1 which

he will reveal. This explains the other two Nash equilibria whereby the prosecutor reveals only α_1 and the lawyer either reveals all his arguments and expects the prosecutor to defeat α_2 or mixes between revealing his two arguments equally.

Expectedly, it is obvious from the graphs below that both WPL and GIGA are converging to the mixed Nash equilibrium $[(0, \frac{1}{2}, \frac{1}{2}, 0)_{p1}, (0, \frac{1}{2}, \frac{1}{2}, 0)_{p2}]$. However, when δ is 0.002 both agents' policies oscillate between (0,1,0,0) and (0,0,1,0), GIGA more ?aggressively? than WPL, as seen in figures 4.9 and 4.11. This is because if we take out the dominated strategies the game is focused around two actions per agent as shown in the sub-game in table 4.5 which has only one mixed Nash equilibrium $[(\frac{1}{2}, \frac{1}{2})_{p1}, (\frac{1}{2}, \frac{1}{2})_{p2}]$. Initially each of the players want to maximize his utility so for example player 1 will start to move toward a pure policy (0,1) in order to try to get a utility of 2. Player 2 will notice this and try to maximize his own utility based on player 1's choice to reveal α_2 by revealing α_1 and going toward (0,1) which will give him a utility of 1. Player 1 will in turn start getting utility 0 and will learn that if player 1 is revealing α_1 then he should start revealing α_4 and change his policy to (1,0) to get utility 1. Again player 2 will start realizing that if player 1 is revealing α_4 he can increase his utility to 2 if he changes his policy to (1,0). And the cycle continues.

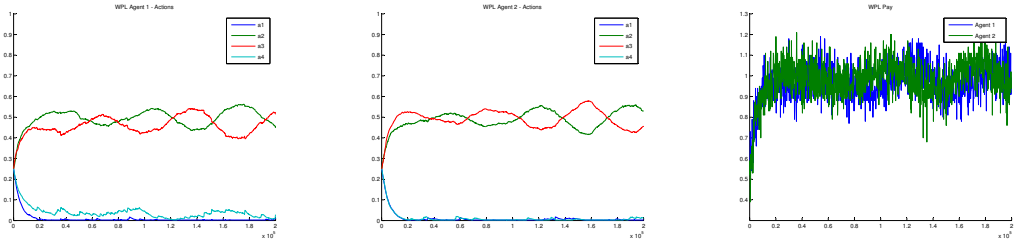
The update equation of WPL dampens these oscillations because the rate of increase of probability toward choosing the better action decreases as that probability increases so it prevents the fast, intuitively more haste, move toward the better action that GIGA does.

	$\{\alpha_3\}$	$\{\alpha_1\}$
$\{\alpha_4\}$	1 2	1 1
$\{\alpha_2\}$	2 0	0 1

Table 4.5: Game 2 Sub-game

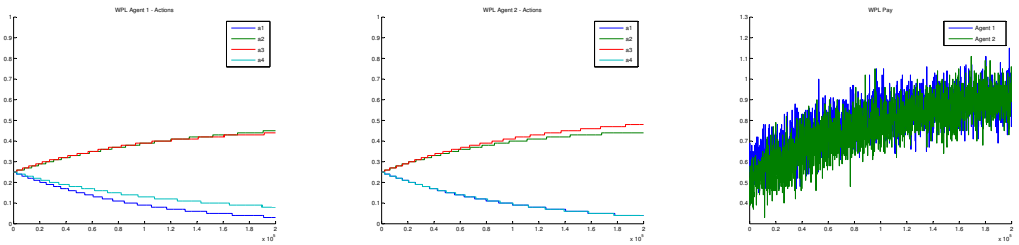
When δ is decreased to 0.0001 the algorithms do not oscillate anymore because the steps to update the policy are small and so both move slower and stabilize at the mixed Nash equilibrium before either player gets a chance to try to deviate toward a pure policy as seen in figures 4.10 and 4.12. GIGA converges faster and WPL might just need more iterations of the game, or time steps, because in addition to the policy learning rate being really small, it is more cautious and so it takes more time to increase enough to reach the equilibrium policy and stabilize.

This accurately mimics the behavior of a lawyer a prosecutor in real life which usually entails formulating arguments that, regardless of having a true basis, are not completely true and build on what the opponent reveals. The agents in this simulation base their choice of actions on the goal of maximising their utilities based on the other agent's policy.



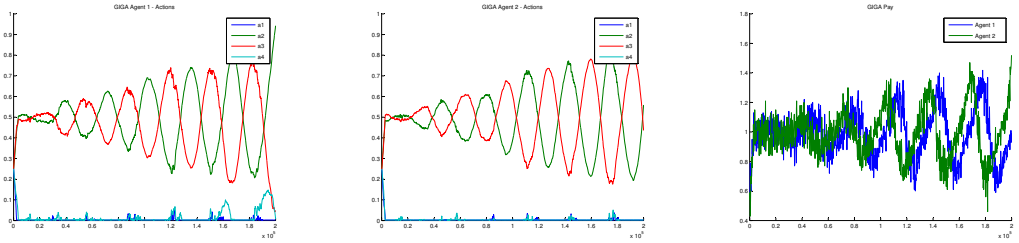
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.9: Game 2: WPL α=0.01 δ=0.002 ϵ=0.01



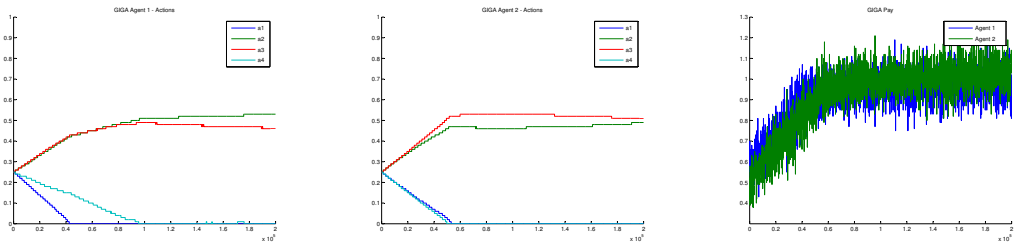
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.10: Game 2: WPL α=0.01 δ=0.0001 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.11: Game 2: GIGA α=0.01 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.12: Game 2: GIGA α=0.01 δ=0.0001 ϵ=0.01

4.3.3 Case 3: Indirect Support

John has been ruled guilty of the bank robbery. In a segment of the court hearing the issue is determining when he did it, based on which his sentence will be issued.

The prosecutor is claiming, and wants to prove, that it was carried out on Monday, during daytime, while the bank was open for business, while John's lawyer wants to prove that it was carried out on Monday, which was Bank Holiday, at night. The sentence for a night robbery on a holiday would not be as harsh because the bank is closed all day, and due to decreased visibility at night time and less passers-by which means less planning and mischief involved, it is easier. From the prosecutor's point of view, the goal is to prove that the robbery occurred on Monday during the day but depending on the lawyer's stance, he can either argue that it was early morning or at noon. Proving that it happened early morning by saying that this is when the evidence was found and the robbery was reported to the police would probably get John a harsher sentence since that means that most of the bank staff were there rather than him taking advantage of some being on lunch break but the lawyer can counter argue that by

stating that Monday was a bank holiday and the bank was closed on that day. According to the lawyer, a stronger argument to make would be that the bank was closed on Monday, since it overrules the prosecutor's claim that the staff were present, but he places more importance on winning the argument that the evidence was found on Monday night, which is what he is interested in proving.

The argumentation graph representing this case, as shown in figure 4.13, is similar to that of case 2 above but without the contradicting, or mutually defeating, arguments. For each agent only one of the arguments given to it defeats the other one. The arguments are as follows:
v: "The day of the robbery was Monday which is a Bank Holiday"
p: "The bank was open for business on the day of the robbery"
q: "There was staff in the building at the time of the robbery"
r: "The robbery occurred at night time"
s: "The robbery occurred during lunchtime"

Leading to the following explicit arguments:
α_1: $v, v \rightarrow \neg p \vdash \neg p$
"The day of the robbery is Monday which is Bank Holiday, the bank is closed on Bank Holiday. Hence the bank was closed on the day of the robbery"
α_2: $p, p \rightarrow q \vdash q$
"The bank was open on the day of the robbery, when the bank is open there is staff in the building. Hence there were staff in the building"
α_3: $(p \wedge \neg q), (p \wedge \neg q) \rightarrow r \vdash r$
"The bank was open on the day of the robbery and there were no staff in the building at the time, on days when the bank is open for business the staff are only there during the day. Hence the robbery happened at night after business hours"
α_4: $\neg q, \neg q \rightarrow s \vdash s$
"There were no staff at the time of the robbery, the staff leave the building to go for lunch during lunch time. Hence the robbery happened during lunch time"

Summarized to the following:
α_1: "The day of the robbery is Monday which is Bank Holiday so the bank was closed"
α_2: "The bank was open on the day of the robbery so there were staff in the building"
α_3: "The bank was open on the day of the robbery and there were no staff in the building at the time so it happened at night after business hours"
α_4: "There were no staff at the time of the robbery so it happened during lunch time"

The lawyer has arguments α_1 and α_3 with utilities of 1 and 5 respectively, his main goal is to prove that the robbery occurred at night. The prosecutor has arguments α_2 and α_4 with utilities 4 and 1 respectively since he places higher priority on winning α_2 but would settle for winning α_4 in case α_2 was defeated by the lawyer.

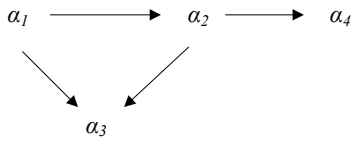


Figure 4.13: Game 3 - Arg Graph

Player	Argument (Utility)
p0	α_1 (1), α_3(5)
p1	α_2 (4), α_4(1)

Table 4.6: Game 3

	{}	$\{\alpha_4\}$	$\{\alpha_2\}$	$\{\alpha_2,\alpha_4\}$
{}	0 0	0 4	0 4	0 4
$\{\alpha_3\}$	5 0	5 1	0 4	0 4
$\{\alpha_1\}$	1 0	1 1	1 0	1 1
$\{\alpha_1,\alpha_3\}$	1 0	1 1	1 0	1 1

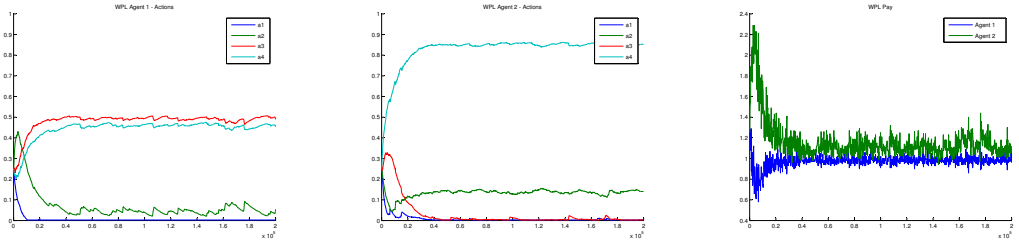
Table 4.7: Game 3 Representation

In the representation of the game in table 4.7 it can be deduced that for player 1, representing the lawyer, (1,0,0,0) is strictly dominated by (0,0,1,0) and (0,0,0,1) and weakly dominated by (0,1,0,0). Hence, the lawyer is always better off revealing something rather than staying quiet: either revealing only α_3 and hiding α_1 which defeats it or revealing α_1 are the best options. In other words either using the argument that the robbery occurred on a holiday which means the bank was closed or the argument that there were no staff in the bank at the time of the robbery so it occurred at night time. Revealing both arguments, in which case α_3 will be defeated by α_1 and α_1 will be won is the same as revealing only α_1 since there is no cost to revealing arguments or punishment for revealing arguments that get rejected. For player 2, representing the prosecutor, (1,0,0,0) is strictly dominated by (0,1,0,0) and (0,0,0,1),in addition, (0,1,0,0) and (0,0,1,0) are both weakly dominated by (0,0,0,1). This means the prosecutor is also better off saying something than staying quiet and getting 0 utility but, on the other hand, he is better revealing both his arguments because α_2 is the argument he prefers to win, and it has higher utility than α_4. Therefore, he reveals both arguments and if α_2 is undefeated he wins it, otherwise it is defeated and he wins α_4. He can also play the mixed strategy $(0, \frac{1}{5}, 0, \frac{4}{5})$ whereby one fifth of the time he hides α_2 since that is how much weight is given to hiding it. The following Nash equilibria result:

- $[(0,0,1,0)_{p1}, (0,0,0,1)_{p2}]$
- $[(0,0,0,1)_{p1}, (0,0,0,1)_{p2}]$
- $[(0,0,1,0)_{p1}, (0, \frac{1}{5}, 0, \frac{4}{5})_{p2}]$
- $[(0,0,0,1)_{p1}, (0, \frac{1}{5}, 0, \frac{4}{5})_{p2}]$

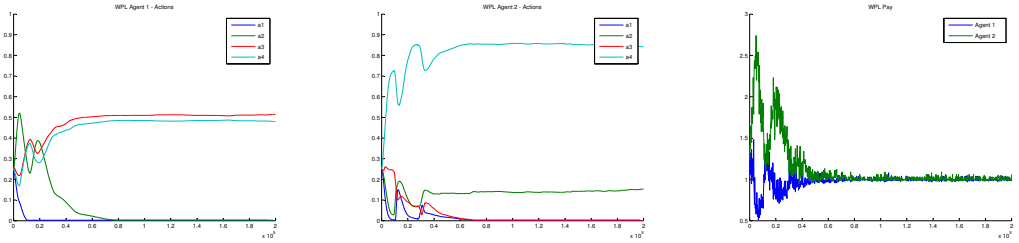
Neither of the learning algorithms converge to any of the Nash equilibria but, interestingly so, each one converges to a combination of two of the equilibria. Using both algorithms, Player 1's policy converges to $(0, 0, \frac{1}{2}, \frac{1}{2})$ which is a combination of the dominant strategies (0,0,1,0) and (0,0,0,1). This is because as mentioned above, from the lawyer's perspective, revealing both arguments is the same as revealing only α_1 in that both actions will get him a guaranteed utility of 1 so mixing between the two actions in any proportion will also get him a utility of 1. As for the prosecutor, WPL converges to the mixed strategy $(0, \frac{1}{5}, 0, \frac{4}{5})$ where as GIGA converges to the pure strategy strategy (0,0,0,1). This also confirms previous results that WPL prefers mixed strategies. In both cases the pay of both agents converges to 1.

When α is 0.001 WPL oscillates for a little bit in the beginning but eventually converges leading to an even smoother graph than when α is 0.01 as seen in in figures 4.14 and 4.15. GIGA's performance, on the other hand, is better when α is 0.01 as seen in in figures 4.16 and 4.17.



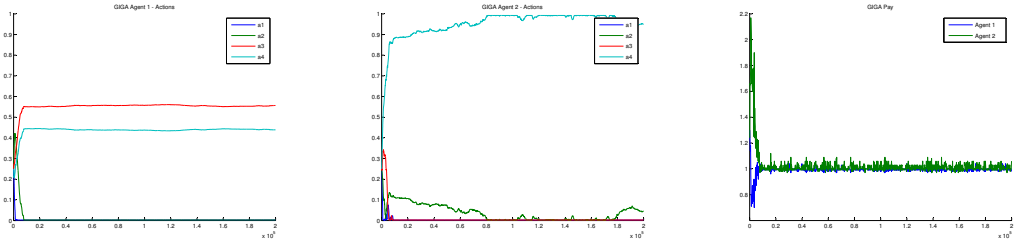
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.14: Game 3: WPL α=0.01 δ=0.002 ϵ=0.01



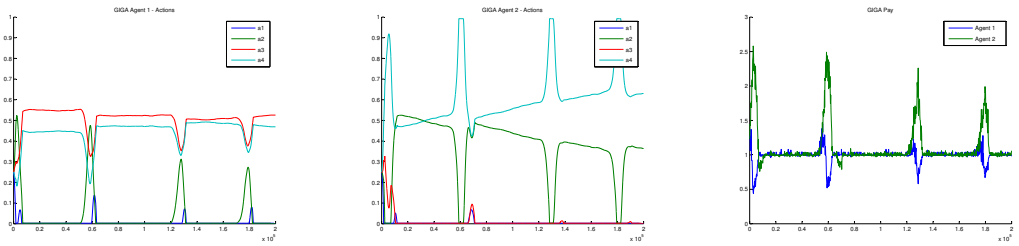
(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.15: Game 3: WPL α=0.001 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.16: Game 3: GIGA α=0.01 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.17: Game 3: GIGA α=0.001 δ=0.002 ϵ=0.01

4.3.4 Case 4: "Argumentative Battle of The Sexes" Game

This game is based on the well known *Battle of The Sexes* game [22] with an argumentative element [20]. A couple is planning their day, in other words arguing, in an attempt to decide on whether they will go to a soccer match or to the ballet. They use the following arguments:
α_1: "We should go to the soccer"
α_2: "We should go to the ballet"
α_3: "Alice is too sick for the outdoors"
α_4: "Brian's ex-wife will be at the ballet"
Brian, the husband, would like to go to the soccer, and states that using α_1, while Alice, the wife, prefers to go to the ballet, and states that using α_2. Therefore α_1 and α_2 attack each other since, due to time constraints, they can only go to one. The other alternative for each one of them is less desirable hence Alice also has the argument α_3 that she is too sick for the outdoors, which attacks α_1, and Brian also has the argument α_4 that his ex-wife will be at the ballet, which attacks α_2. The least desirable option for both of them is to not go anywhere, so

each of them would still rather go to the other's choice rather than stay at home. The resulting argumentation graph to represent this game is shown in figure 4.18.

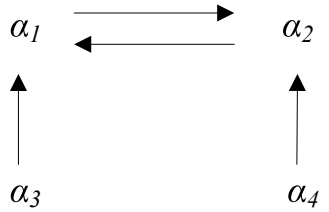


Figure 4.18: Game 4 - Arg Graph

Player	Argument (Utility)
p0	α_2 (1), α_3(1)
p1	α_1 (1), α_4(1)

Table 4.8: Game 4

	{}	{α_4}	{α_1}	{α_1,α_4}
{}	0 0	0 1	0 1	0 2
{α_3}	1 0	1 1	1 0	1 1
{α_2}	1 0	0 1	0 0	0 2
{α_2,α_3}	2 0	1 1	2 0	1 1

Table 4.9: Game 4 Representation

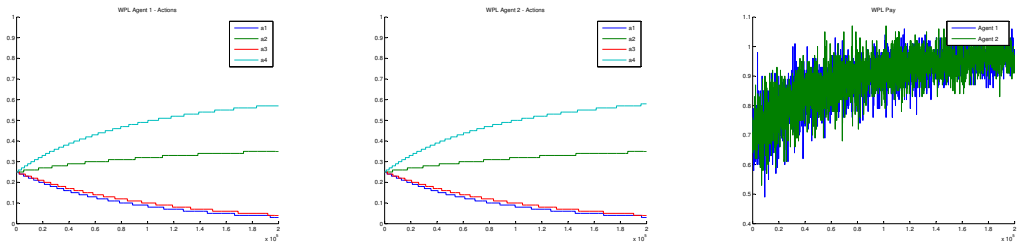
When each of the arguments is given an equal utility of one as shown in table 4.8, the resulting game representation that is formulated by the argumentation game simulator, shown in table 4.9, is a little bit different from the intuition of the original game. Here, the utility of winning both arguments is the sum of utilities of winning each argument, since there is no provision for assigning utilities for winning combinations of arguments in the current version of the simulator, and so the utility gained by Brian and Alice is related to the number of arguments won by each and not where they end up going. However, the game in this form is worth examining before moving on to the original game.

The strategies of Alice and Brian turn out to be symmetric because of the symmetric nature of the graph, for both of them (1,0,0,0) and (0,0,1,0) are strictly dominated strategies and (0,1,0,0) is weakly dominated by (0,0,0,1). In other words, for both of them staying quiet or only saying their first option is definitely the worst they can do and only arguing against the other's choice is better but not as good as using both arguments. This game has four Nash equilibria: Nash Equilibria:

- $[(0,1,0,0)_{p1},(0,1,0,0)_{p2}]$
- $[(0,1,0,0)_{p1},(0,0,0,1)_{p2}]$
- $[(0,0,0,1)_{p1},(0,1,0,0)_{p2}]$
- $[(0,0,0,1)_{p1},(0,0,0,1)_{p2}]$

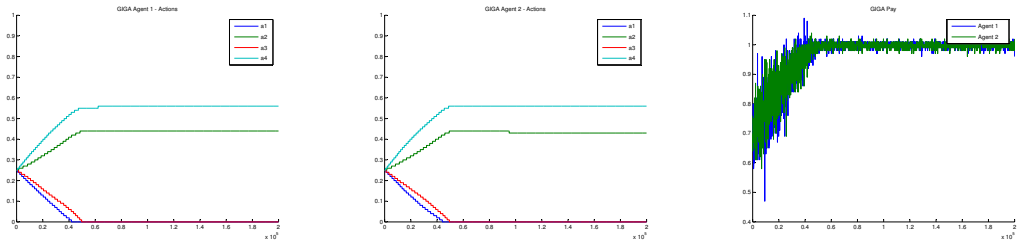
which are the four possible combinations of joint strategies between each saying only where they would like to go or revealing both arguments, in other words saying their preference but also using the other argument to attack the other's choice. All four Nash equilibria gain each one of them a utility of one.

As seen in figures 4.19 and 4.20, WPL does almost converges to $[(0, 0.37, 0, 0.63)p1, (0, 0.37, 0, 0.63)p2]$ and GIGA to $[(0, 0.45, 0, 0.55)p1, (0, 0.45, 0, 0.55)p2]$ and both with the pay converging to (1,1). This is because interestingly any combination of the two strategies (0,1,0,0) and (0,0,0,1) by both players will lead to a pay of (1,1).



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.19: Game 4: WPL α=0.01 δ=0.0001 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.20: Game 4: GIGA α=0.01 δ=0.0001 ϵ=0.01

As mentioned earlier, the utilities in the original game are dependent on where the couple ends up going. For Brian the utility of going to the soccer match is the highest followed by going to the ballet followed by staying at home. For Alice, on the other hand, the utility of going to the ballet is the highest followed by going to the soccer game followed by staying at home. So a utility of two is assigned to each of them for their first choice, one for the second choice and zero for staying at home. There are three cases where Alice gets her way, and they

go to the ballet: she either says that she want to go to the ballet and Brian stays quiet or she says that and also attacks the soccer match option, in anticipation of his argument, and he either stays quiet or only suggests going to the soccer match but does not use the argument that his ex-wife will be at the ballet. Similarly there are three cases where Brian gets his way and they end up going to the soccer match. The rest of the cases they end up staying at home. The representation of this game which was examined as a generic 2-player 4-action game in the simulator is shown in table 4.10.

	{}	$\{\alpha_4\}$	$\{\alpha_1\}$	$\{\alpha_1,\alpha_4\}$
{}	0 0	0 0	1 2	1 2
$\{\alpha_3\}$	0 0	0 0	0 0	0 0
$\{\alpha_2\}$	2 1	0 0	0 0	1 2
$\{\alpha_2,\alpha_3\}$	2 1	0 0	2 1	0 0

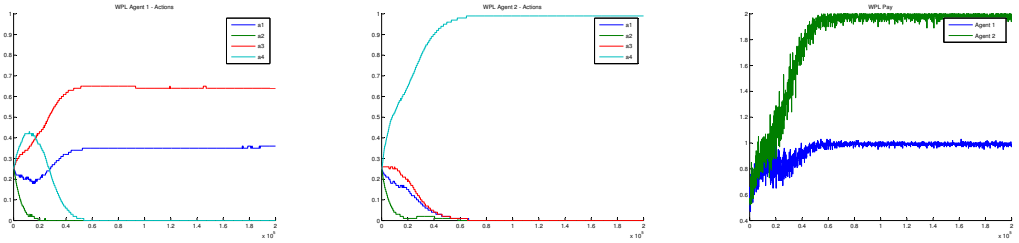
Table 4.10: Game 4 Representation - Original Game

For both husband and wife (0,1,0,0), which is to only use the argument that attack's the partner's suggestion, is a weakly dominated strategy obviously because playing it will guarantee staying at home whatever the other plays and the following Nash equilibria result:

- $[(0,1,0,0)_{p1},(0,1,0,0)_{p2}]$
- $[(1,0,0,0)_{p1},(0,0,0,1)_{p2}]$
- $[(0,0,1,0)_{p1},(0,0,0,1)_{p2}]$
- $[(0,0,0,1)_{p1},(1,0,0,0)_{p2}]$
- $[(0,0,0,1)_{p1},(0,0,1,0)_{p2}]$
- $[(1,0,0,0)_{p1},(0,0,\frac{1}{2},\frac{1}{2})_{p2}]$
- $[(\frac{1}{7},0,\frac{2}{7},\frac{4}{7})_{p1},(\frac{1}{7},0,\frac{2}{7},\frac{4}{7})_{p2}]$
- $[(0,0,\frac{1}{2},\frac{1}{2})_{p1,(}\,1,0,0,0)p2]$

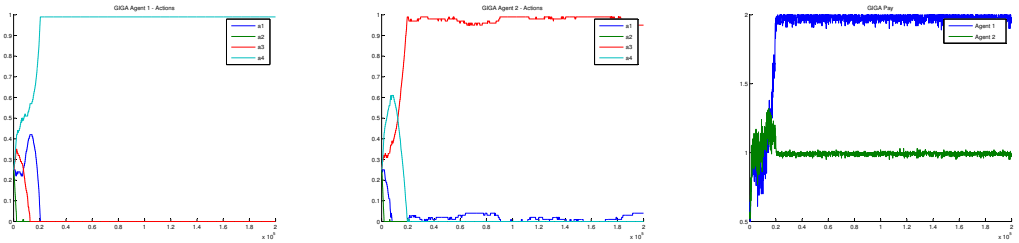
Interestingly enough though their joint strategy when they both play, and end up staying at home with a joint utility of (0,0), is a Nash equilibrium. The mixed strategy Nash equilibrium $[(\frac{1}{7},0,\frac{2}{7},\frac{4}{7})_{p1},(\frac{1}{7},0,\frac{2}{7},\frac{4}{7})_{p2}]$ gets them a joint utility of $(\frac{6}{7},\frac{6}{7})$ which means that they end up staying at home some of the time. The rest of the Nash equilibria are Pareto optimal and ensure that the couple ends up going somewhere leading to a higher utility of either (1,2), if Brian wins and they go to the soccer match, or (2,1), if Alice is more convincing and they go to the ballet. This , of course happens if either one of them stays quiet and the other at least suggests his/her idea and possibly argues against the other's , or mixes between these two strategies (0,0,1,0) and (0,0,0,1), so whoever speaks out gets their way. It also happens when one of them only makes an initial suggestion while the other attacks it and makes his/her suggestion in which case the one who uses both arguments gets his/her way over the one who used only one.

As seen in figures 4.21 and 4.22, when α=0.01 and δ=0.002 WPL converges to the strategy $[(\frac{1}{3},0,\frac{2}{3},0)_{p1,(}\,0,0,0,1)p2]$ and pay (1,2) which means they go to the soccer match, and GIGA converges to $[(0,0,0,1)_{p1,(}\,0,0,1,0)p2]$ and pay (2,1) which means they go to the ballet. Again, WPL's bias against pure strategies causes the agent representing Alice to mix between the strategies (1,0,0,0) and (0,0,1,0) proportionally to their expected payoff of 2 and 3 respectively but also maximizing the collective pay of both agents and is therefore Pareto optimal.



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.21: Game 4 Original Game: WPL α=0.01 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Pay

Figure 4.22: Game 4 Original Game: GIGA α=0.01 δ=0.002 ϵ=0.01

4.3.5 Case 5: The Floating Defeater Game

This game simulates a segment of a court hearing where John is appealing the ruling convicting him of committing a murder. The prosecutor instigating that John is guilty of committing the murder is present along with John's two defenders who are have conflicting reasons for his innocence but they are both interested in the judge pronouncing him innocent and defeating the prosecutor's accusation. One of them claims, and has proof, that John was not present at the crime scene but rather at his brother's house at the time of the murder. The other also claims, and has proof, that John was at the park during the execution of the crime and he was the person who called the police and reported it as a witness.

The Floating Defeater game is adopted from a characteristic argumentation graph that is used in argumentation literature to demonstrate various concepts in argumentation [17]. As seen in figure 4.23, α_1 and α_2 defeat each other and they both defeat α_3 which itself defeats α_4.

Given that:

r: "John was at his brother's house at the time of the murder"
q: "John was at the crime scene at the time of the murder"
v: "John is a witness of the murder"
p: "John committed the murder"
s: "John is the murderer"
t: "John's sentence should be lifted"

Then the following explicit arguments can be constructed as follows:
α_1: $r, r \rightarrow \neg q \vdash \neg q$
"John was at his brother's house, if he was at his brothers house then he was not at the park. Hence he was not at the park where the crime scene was, at the time of the murder"
α_2: $v, v \rightarrow (q \wedge \neg p) \vdash (q \wedge \neg p)$
"John witnessed the murder, a witness is usually present at the crime scene but is not the criminal. Hence he was at the crime where at the time of the murder but he did not perform it"
α_3: $(q \wedge p), (q \wedge p) \rightarrow s \vdash s$
"John was at the crime scene at the time of the murder and he committed the murder, the person who was present at the crime scene and commits the murder is a murderer. Hence he is a murderer"
α_4: $\neg s, \neg s \rightarrow t \vdash t$
"John is not a murderer, a person who is not the murderer should not be sentenced. Hence his sentence should be lifted"

Hence, the arguments in the graph are an abstraction of the following statements:
α_1: "John was at his brother's house, and therefore not at the park where the crime scene was, at the time of the murder"
α_2: "John witnessed the murder therefore he was at the crime where at the time of the murder but he did not perform it"
α_3: "John was at the crime scene at the time of the murder and he committed the murder, therefore he is a murderer"
α_4: "John is not a murderer and therefore his sentence should be lifted"

There are three agents in this game. Agent 1, representing the first defender, has arguments α_1 and α_4, agent 2, representing the second defender, has arguments α_2 and α_4 and agent 3, the prosecutor, has argument α_3 out of which they can reveal or hide. Intuitively, arguments α_1 and α_2 mutually defeat each other (it is impossible for John to be at two places at the same time) and they both defeat argument α_3 since they both state facts that contradict the possibility of John being the murderer. Hence, John's innocence will be compromised if both α_1 and α_2 were revealed since neither one will be won and instead α_3 will be undefeated and will be won by the prosecutor.

It is important to note that none of the agents in the game are self-defeating, i.e. have arguments that defeat each other, so each agent's Pareto optimal strategy should be to reveal all his arguments since they have utility maximizing preferences [19].

Argument α_3 defeats argument α_4 but not the other way around. This is because the defendant is already convicted of the crime and this is an appeal so simply claiming his innocence will not overrule the conviction whereas the court ruling stating that he is guilty does defeat any claim of his innocence.

Player 1 and player 2, representing the two lawyers, have two arguments, and hence four actions, each. P1 can choose to reveal {}, {α_4}, {α_1} or {α_1,α_4} and P2 can choose to reveal {}, {α_4}, {α_2} or {α_2,α_4}. Player 3, representing the prosecutor, has only one argument and so he can either hide it by choosing action {} or reveal it by choosing action {α_3}. The strategic game representation of this game is best illustrated in the form of two tables, one when player 3 hides his argument and the other when he reveals it. For each action combination chosen amongst the three players the corresponding utilities are given for p1, p2 and p3 respectively.

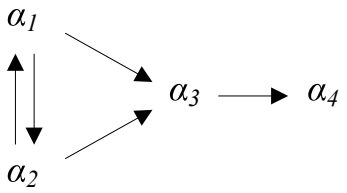


Figure 4.23: Game 5 - Arg Graph

Variant 1: Collaborative

This variant of the game is a case where the two lawyers are equally interested in John's innocence but have no motivation to win their respective backing arguments. In other words, the utility to the defendants of winning α_1 and α_2 respectively is 0 and so they are indifferent toward the reason for John's innocence or who's argument proves it but winning α_4 is given a utility of 1 for both. A utility of 1 is given to α_3 for the prosecutor.

Player	Argument (Utility)
p1	α_1 (0), α_4 (1)
p2	α_2 (0), α_4 (1)
p3	α_3 (1)

Table 4.11: Game 5 Variant 1

Intuitively, in this variant of the game, one would expect that the two lawyers would eventually adopt a strategy whereby one would reveal his backing argument while the other withholds his, in order to get it to be accepted and defeat the prosecutor's argument, and both would claim John's innocence for utility. The prosecutor is expected to reveal his argument all the time since that will maximize his utility.

As seen in the strategic game translation in table 4.12, the two lawyers, represented by p1 and p2, are in exactly the same situation in terms of the best strategy to maximize their respective utilities. Staying quiet, by hiding both arguments and choosing action {}, or revealing only the backing argument, by choosing $\{\alpha_1\}$ or $\{\alpha_2\}$ respectively, will get each lawyer 0 utility because they each only care about winning the final argument $\{\alpha_4\}$. Hence, revealing α_4 is the only chance at gaining any utility for each of the lawyers and so the two strategies (1,0,0,0) and (0,0,1,0) are weakly dominated by (0,1,0,0) and (0,0,0,1).

As for the prosecutor, p3, whose aim is obviously to prove that John is guilty of the murder, out of his two actions staying quiet is weakly dominated by revealing his argument α_3 which, if undefeated, will get him a utility of 1.

Additionally, revealing all arguments is indeed a Pareto optimal strategy for each of the agents but not sometimes not a dominant strategy.

(a)

	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 0	0 1 0	0 0 0	0 1 0
{α_4}	1 0 0	1 1 0	1 0 0	1 1 0
{α_1}	0 0 0	0 1 0	0 0 0	0 1 0
{α_1,α_4}	1 0 0	1 1 0	1 0 0	1 1 0

(b)

	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 1	0 0 1	0 0 0	0 1 0
{α_4}	0 0 1	0 0 1	1 0 0	1 1 0
{α_1}	0 0 0	0 1 0	0 0 0	0 0 0
{α_1,α_4}	1 0 0	1 1 0	0 0 0	0 0 0

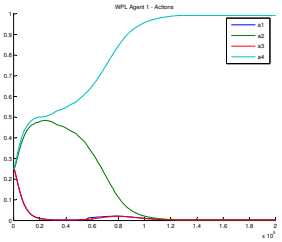
Table 4.12: Game 5 Variant 1 Representation

Therefore, solving this game results in the following Nash equilibria: five pure and one mixed as follows:

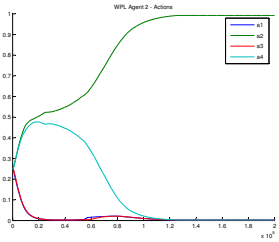
- $[(0,1,0,0)_{p1},(0,0,0,1)_{p2},(1,0)_{p3}]$
- $[(0,0,0,1)_{p1},(0,1,0,0)_{p2},(0,1)_{p3}]$
- $[(0,0,0,1)_{p1},(0,1,0,0)_{p2},(1,0)_{p3}]$
- $[(0,1,0,0)_{p1},(0,0,0,1)_{p2},(0,1)_{p3}]$
- $[(0,0,0,1)_{p1},(0,0,0,1)_{p2},(1,0)_{p3}]$
- $[(0,\frac{1}{2},0,\frac{1}{2})_{p1},(0,\frac{1}{2},0,\frac{1}{2})_{p2},(0,1)_{p3}]$

The Nash equilibria represent the expected intuitive behavior of the three agents in terms of joint strategies. In the first four Nash equilibria each lawyer either plays (0,1,0,0), reveals only α_4, or plays (0,0,0,1), reveals both his arguments, while the other does the opposite. The prosecutor either hides or reveals. In addition, when the prosecutor hides his argument, both the lawyers can reveal all their arguments and that will be a Nash equilibrium since they don't have to worry about defending the final argument by having an undefeated backing argument to attack the prosecutor's argument. Finally, the last Nash equilibrium is when the prosecutor shows his argument and the lawyers adopt a mixed strategy of revealing their respective backing arguments, along with the final argument, half the time and revealing only the final argument the rest of the time. In other words their joint strategy is to take turns to reveal their backing arguments.

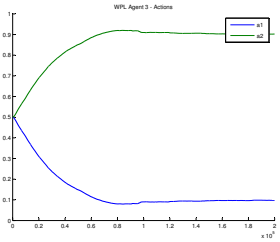
When δ, the policy learning rate, is 0.002, both WPL and GIGA converge to one of the pure Nash equilibria gaining the lawyers a pay of 1 and the prosecutor a pay of 0 as seen in figures 4.24 and 4.25. The strategies of agents 1 and 2, the two lawyers, are more definite but agent 3, the prosecutor, learns to reveal his argument, even though it is defeated and he gets no utility, because there is no cost to doing that. WPL's convergence is more accurate as α decreases, again confirming previous results, and achieving very smooth graphs when α is 0.001. GIGA converges faster than WPL when α is 0.1 and decreasing α only slows it down (refer to Appendix C for additional graphs).



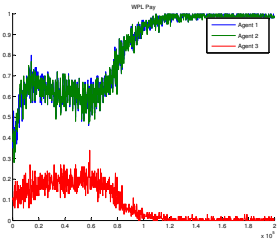
(a) Agent 1



(b) Agent 2

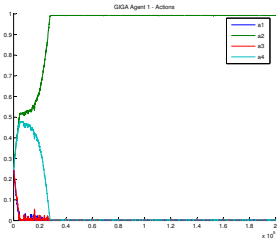


(c) Agent 3

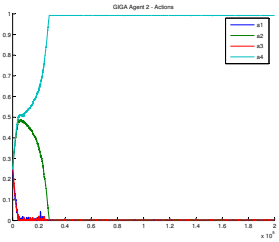


(d) Pay

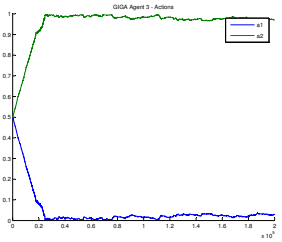
Figure 4.24: Game 5v1: WPL α=0.001 δ=0.002 ϵ=0.01



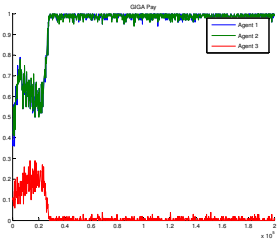
(a) Agent 1



(b) Agent 2



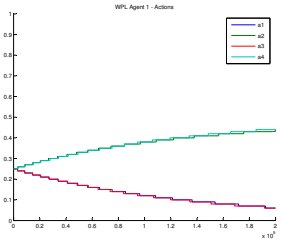
(c) Agent 3



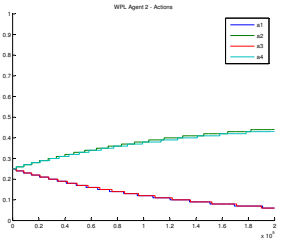
(d) Pay

Figure 4.25: Game 5v1: GIGA α=0.1 δ=0.002 ϵ=0.01

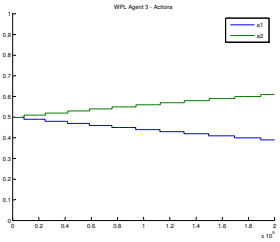
When δ is decreased to 0.0001 the learning algorithms head toward the mixed strategy Nash equilibrium $[(0, \frac{1}{2}, 0, \frac{1}{2})_{p1}, (0, \frac{1}{2}, 0, \frac{1}{2})_{p2}, (0, 1)_{p3}]$ but do not converge. The policies of the agents representing the lawyers do actually converge in GIGA as seen in figure 4.27 but their pay does not converge because the prosecutor's strategy is changing and needs more time to reach (0,1).



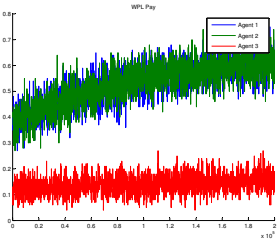
(a) Agent 1



(b) Agent 2

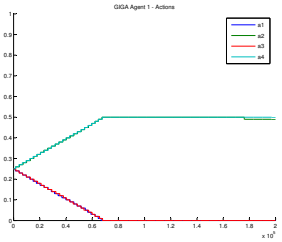


(c) Agent 3

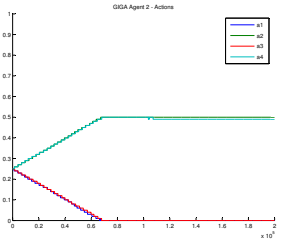


(d) Pay

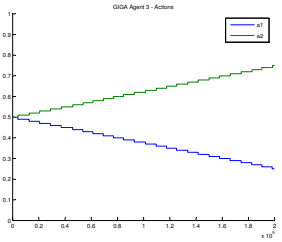
Figure 4.26: Game 5v1: WPL α=0.01 δ=0.0001 ϵ=0.01



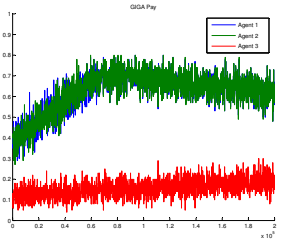
(a) Agent 1



(b) Agent 2



(c) Agent 3



(d) Pay

Figure 4.27: Game 5v1: GIGA α=0.01 δ=0.0001 ϵ=0.01

Variant 2: Self-interested

In this variant of the game the lawyers are self interested in the sense that they are more interested in winning their backing arguments than in proving John's innocence. This would be the case if the two lawyers were competing, for example, for winning the case in order to build a reputation for themselves rather than simply for the sake of John. Therefore for the two agents representing the lawyers α_1 and α_2 are given a utility of 5 and α_4 is given a utility of 2. The prosecutor's stance is the same as the first variant and so a utility of 1 is given to α_3.

Player	Argument (Utility)
p1	α_1 (5), α_4 (2)
p2	α_2 (5), α_4 (2)
p3	α_3 (1)

Table 4.13: Game 5 Variant 2

The only difference between this variant of the game and the previous one is that in this case there is a utility assigned to the agents representing the lawyers for winning the backing arguments. From the strategic game representation shown in table 4.14 it can be deduced that

again for these two agents (1,0,0,0) is weakly dominated by all the other strategies and (0,0,1,0) by (0,0,0,1) but still revealing all arguments is a Pareto optimal strategy.

(a)

	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 0	0 2 0	0 5 0	0 7 0
{α_4}	2 0 0	2 2 0	2 5 0	2 7 0
{α_1}	5 0 0	5 2 0	0 0 0	0 2 0
{α_1,α_4}	7 0 0	7 2 0	2 0 0	2 2 0

(b)

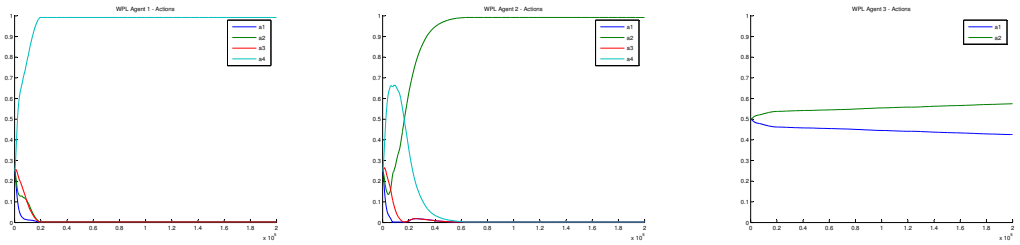
	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 1	0 0 1	0 5 0	0 7 0
{α_4}	0 0 1	0 0 1	2 5 0	2 7 0
{α_1}	5 0 0	5 2 0	0 0 0	0 0 0
{α_1,α_4}	7 0 0	7 2 0	0 0 0	0 0 0

Table 4.14: Game 5 Variant 2 Representation

The Nash Equilibria are expectedly also similar to those of the first variant except for the mixed one where in this variant each lawyer will opt to reveal his backing argument more than half the time, specifically seven out of nine times, because winning his backing argument is of considerable value to him:

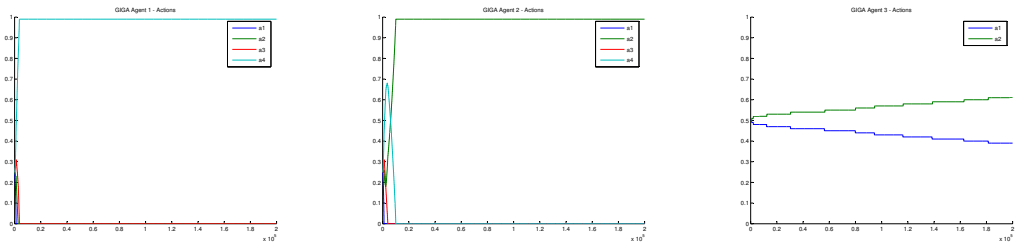
- $[(0,1,0,0)_{p1},(0,0,0,1)_{p2},(1,0)_{p3}]$
- $[(0,0,0,1)_{p1},(0,1,0,0)_{p2},(0,1)_{p3}]$
- $[(0,0,0,1)_{p1},(0,1,0,0)_{p2},(1,0)_{p3}]$
- $[(0,1,0,0)_{p1},(0,0,0,1)_{p2},(0,1)_{p3}]$
- $[(0,0,0,1)_{p1},(0,0,0,1)_{p2},(1,0)_{p3}]$
- $[(0,\frac{2}{9},0,\frac{7}{9})_{p1},(0,\frac{2}{9},0,\frac{7}{9})_{p2},(0,1)_{p3}]$

As seen in figures 4.28 and 4.29 the policies of the lawyers in both WPL and GIGA converge very fast to the pure Nash equilibrium $[(0,0,0,1)_{p1},(0,1,0,0)_{p2},(0,1)_{p3}]$ and the prosecutor starts to head toward it but needs more time to reach it. This means that the lawyers quickly learn to collaborate and one of them learns that he should hide his backing argument and let the other one defend α_4 using his argument and settle for a utility of 2, which is not as good as 5 but better than 0.



(a) Agent 1 (b) Agent 2 (c) Agent 3

Figure 4.28: Game 5v2: WPL α=0.001 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Agent 3

Figure 4.29: Game 5v2: GIGA α=0.001 δ=0.002 ϵ=0.01

Variant 3: Compromising

In this variant of the game one of the lawyers is self-interested and cares more about winning his backing argument rather than in John's innocence while the other actually prefers that his backing argument is not accepted. However, they both equally care for John's innocence. Hence the second defender prefers that the other's backing argument, rather than his own, is the one that is accepted in order to defend John's innocence. This would be the case, for example, if the second lawyer was being forced to defend John, even though he is not convinced of his innocence himself, but he is willing to lie in order to win the case and get his compensation. Therefore agent 1, representing the first lawyer, gets a utility of 5 for winning α_1 while agent 2, representing the second lawyer, gets a utility of -5 for winning α_2 and both get a utility of 2 for winning α_3. Again, a utility of 1 is given to the prosecutor's argument α_3.

Player	Argument (Utility)
p1	α_1 (5), α_4 (2)
p2	α_2 (-5), α_4 (2)
p3	α_3 (1)

Table 4.15: Game 5 Variant 3

In this variant since the second lawyer prefers not to use his backing argument, and he is assigned a negative utility to winning α_2, intuitively he should avoid revealing it. As expected, through analysis of the game representation in table 4.16, it can be deduced that for the first agent (1,0,0,0) is weakly dominated bu all the other strategies and (0,0,1,0) by (0,0,0,1) while for the second agent (1,0,0,0) is weakly dominated by (0,1,0,0) and (0,0,1,0) is strongly dominated by (0,1,0,0) making (0,1,0,0) is the weakly dominant strategy. Once again, revealing all arguments is a Pareto optimal strategy for each of the agents.

(a)

	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 0	0 2 0	0 -5 0	0 -3 0
{α_4}	2 0 0	2 2 0	2 -5 0	2 -3 0
{α_1}	5 0 0	5 2 0	0 0 0	0 2 0
{α_1,α_4}	7 0 0	7 2 0	2 0 0	2 2 0

(b)

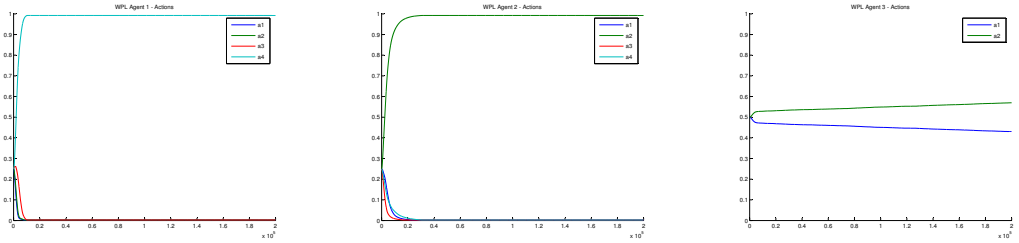
	{}	{α_4}	{α_2}	{α_2,α_4}
{}	0 0 1	0 0 1	0 -5 0	0 -3 0
{α_4}	0 0 1	0 0 1	2 -5 0	2 -3 0
{α_1}	5 0 0	5 2 0	0 0 0	0 0 0
{α_1,α_4}	7 0 0	7 2 0	0 0 0	0 0 0

Table 4.16: Game 5 Variant 3 Representation

The first two Nash equilibria reflect the expected behavior of the agents whereby the first lawyer will reveal all his arguments, thus being the one who reveals the backing argument, while the other lawyer compromises and only reveals α_4.

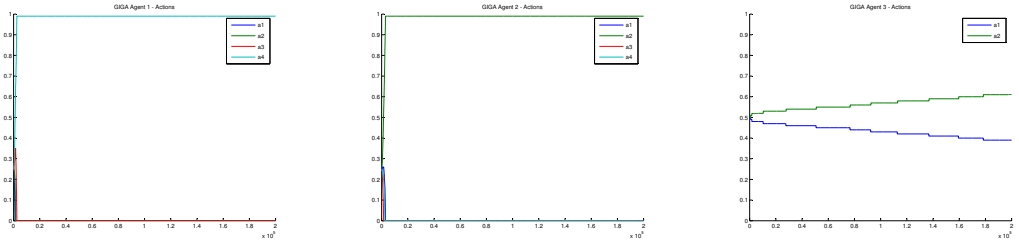
- $[(0,0,0,1)_{p1}, (0,1,0,0)_{p2}, (1,0)_{p3}]$
- $[(0,0,0,1)_{p1}, (0,1,0,0)_{p2}, (0,1)_{p3}]$
- $[(0,0,0,1)_{p1}, (0,0,0,1)_{p2}, (1,0)_{p3}]$

Both learning algorithms converge to the same pure Nash equilibrium $[(0,0,0,1)_{p1}, (0,1,0,0)_{p2}, (0,1)_{p3}]$ as in variant 2 as seen in figures 4.30 and 4.31 for the two lawyers but in this case they converge faster. This is because player 2, representing the second lawyer, actually prefers hiding his backing argument.



(a) Agent 1 (b) Agent 2 (c) Agent 3

Figure 4.30: Game 5v3: WPL α=0.001 δ=0.002 ϵ=0.01



(a) Agent 1 (b) Agent 2 (c) Agent 3

Figure 4.31: Game 5v3: GIGA α=0.001 δ=0.002 ϵ=0.01

4.4 Discussion and Conclusion

Each of the five cases discussed in this chapter contributed to the answers that this experimental study intended to answer, some of which are common amongst all of the cases and some specific to the game.

It can be concluded that the dominance of some strategies over others and the different kinds of Nash equilibria in each game drove the agents to have preferences over the different available strategies and in turn use the more effective strategies to maximize their utility and influence the judge. Indeed in some cases hiding arguments, i.e. lying, as part of a strategy will be preferred by agents. The exception, which was proved by Rahwan and Larson [19], is under the condition the agents are not self defeating in which case the game is strategy proof and the dominant strategy equilibrium is to reveal all the arguments, is apparent in cases 1, 4 and 5. The agents in these cases actually do learn to reach the dominant strategy equilibrium in some cases. In others, however they learn to play strategies that will maximize their payoff, equal

to or even greater than that at a dominant strategy equilibrium if one exists and this is due to the nature of the learning algorithms inherent in the agents as discussed in detail in each case.

As for learning algorithms, the mechanism by which each updates the policy does make it more suited to some types of games. For example WPL's bias toward mixed strategies got the agents in Case 1 a higher joint pay than the dominant strategy Nash equilibrium's pay and its decaying learning rate as it approaches a policy lead to the dampening of the oscillations between the two dominant strategies in Case 2. This is effectively the same as using a smaller policy learning rate, as seen when δ is decreased to 0.0001, but does not slow down the general learning and convergence of the pay. GIGA on the other hand is generally faster than WPL and is biased toward pure strategies.

Chapter 5

Conclusion and Further Work

Argumentation has been studied extensively in the field of Artificial Intelligence, however we know very little about its strategic aspects. This thesis aims to contribute to the general problem of trying to understand strategic behavior in argumentation, over repeated encounters when self-interested agents, in a multi-agent environment, interact. As in any game-theoretic setting, there may not be a single optimal strategy for each agent and so the outcome of the game (i.e. its equilibrium) is not always uniquely predictable. This gives rise to the problem of equilibrium selection [13]. Two typical approaches are usually taken to deal with this problem. The first approach has to do with *equilibrium refinement* [13], that is, coming up with more restricted equilibrium concepts in order to narrow down the possible outcomes of the game. The second approach is by observing convergence through simulation. It is not intuitively clear what variables, or conditions, the agents' choice of equilibria is dependent on and therefore the requirement for quantitative experimentation arises.

Hence the Java-based simulation tool developed by Abdallah and Lesser [1], to examine the same issue through experimentation with different learning algorithms, was extended to implement argumentation games for this purpose and used to run repeated game experiments using combinations of characteristic argumentation games, adapted from literature, and types of adaptive agents under different conditions. The designed experiments and the analysis of the results produced formed the basis for general observations.

The agents use a multitude of different strategies to influence the judge and maximize their pay, thereby revealing different combinations of arguments with different frequencies, depending on the Nash equilibria of the game, the dominance of the pure strategies and the Pareto efficiency of the pure strategies in a game. These are dependent on aspects inherent in the argumentation game such as the argumentation graph, the assignment of arguments to the agents and distribution of utilities over the arguments. While games where the agents were not (directly or indirectly) self-defeating were strategy proof [19], interestingly in the other games, by playing a combination of Nash equilibria in some cases, the agents were able to gain a payoff that is higher than that of any of the individual Nash equilibria. As for the effect of the agent type, in terms of learning algorithms, on the choice of strategy, WPL was found to be biased toward mixed strategies while GIGA was faster in convergence to pure strategy Nash equilibria.

Many questions surface as a result of this initial attempt at exploring these argumentation games. One question worth investigating is since every argumentation game can be represented as a repeated strategic game, can every strategic game be implemented as an argumentation game? Answering this question will involve at least partial mathematical proof.

Other questions such as that of what motivates agents to use certain strategies over others or the characteristics of games that determine the kinds of strategies that result. The answers to these are beyond the scope of this thesis and require extensive experimentation and the extension of the simulation tool to implement more parameters that allow the representation of a wider variety of characteristics in games. This can be done for example through the provision

of combined argument utilities, in other words the total utility of winning multiple arguments is not limited to being the sum of the utilities of winning the individual arguments.

Additionally, aspects pertaining to the type of the "judge" and how strict or unforgiving he/she is can also be easily implemented into the tool. Assigning a cost to revealing arguments, assigning negative utilities for losing arguments and some form of punishment for lying or hiding arguments like exclusion from some iterations of the game or deduction from the utility. More substantial extensions are worthwhile and should also be implemented such as additional acceptability semantics, a graphical user interface to make the input to the game and the display of the results more intuitive.

Appendix A

Technologies and Tools

A.1 Introduction

A.1.1 Java

The argumentation game was developed in Java[1] as an extension to the Java-Based multi-agent learning simulation tool.

A.1.2 Eclipse

The Eclipse SDK[2] is an open source integrated development environment (IDE) that was one of the results of the Eclipse project which was started by IBM in November 2001 and developed into a set of extensible frameworks and tools for building, deploying and managing software across the life cycle. It is primarily used for Java and plug-in development but also supports other programming languages.

The extension of the tool to include the argumentation game was developed using the Eclipse SDK due to both the simple code editing tools it provides, with support for Java syntax, and the automatic compiling feature whereby the classes are compiled as soon as they are saved.

A.1.3 Matlab

Matlab[3] is a high-level technical computing language that provides tools in an interactive environment for mathematical computation algorithm development to allow data visualization and analysis. Matlab was used to write scripts for extracting data from the simulation tool's output log files and plotting the different graphs for visual analysis. It is suited for this purpose due to its powerful and fast data manipulation abilities.

A.1.4 Gambit

Gambit[4] is an open source library of tools with a graphical user interface that allows experimentation with game-theoretic finite extensive and strategic games. It was used for the computation and analysis of the dominant strategies and Nash equilibria of the strategic representation of the argumentation games.

[1] http://java.sun.com/
[2] http://www.eclipse.org/
[3] http://www.mathworks.com/
[4] http://gambit.sourceforge.net/

Appendix B

Sample Game Run

B.1 Introduction

This section will provide a sample game run in order to demonstrate the use of the different input parameters, the layout of a game parameters file and how the output results are read.

B.2 Game Parameters File

The game parameters file *ArgGameParams.txt*, a sample of which is shown in table B.1, is a simple text file that contains the argumentation game parameters listed in section 3.1.3. This is the actual parameters file of the argumentation game studied as Case 3 in Chapter 4.

```
Arguments,A,B,C,D
Links,A,B,B,C,A,C,B,D
Agent,a1,A,1,C,5
Agent,a2,B,4,D,1
```

Table B.1: A sample input parameter file

Each argument in the game is given a letter, in alphabetical order, to represent the name of the argument. The links in the argumentation graph are listed as pairs of from-to defeat relations, so the corresponding four arguments A, B, C and D and four defeat relations: A to B, B to C, A to C and B to D, represent the argumentation graph in figure B.1.

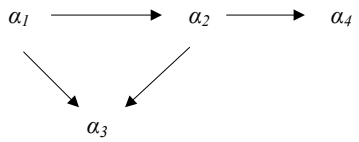


Figure B.1: Sample Argumentation Graph

B.3 Game Output

B.3.1 Game Initiation Output

The following is the console output that reflects the interpretation of contents of the input configuration file *ArgGameParams.txt* and the formulation of the argumentation graph and action sets of the agents.

```
    ARGUMENTATION GAME
Arguments,A,B,C,D
Links,A,B,B,C,A,C,B,D
Agent,a1,A,1,C,5
Agent,a2,B,4,D,1
The Arguments in this game are:
[A, B, C, D]
The Links in this game are:
AB
BC
AC
BD
The Agents in this game are:
[ a1, a2]
The Utilities in this game are:
{a1C=5.0, a2D=1.0, a1A=1.0, a2B=4.0}
init action and pay
init actions for each agent from file
agent a1's action set is
[ [ ], [C], [A], [A, C]]
agent a2's action set is
[ [ ], [D], [B], [B, D]]
init graph from file
Argument A has status nil and defeaters:
Argument B has status nil and defeaters:  A
Argument C has status nil and defeaters:  A B
Argument D has status nil and defeaters:  B
```

B.3.2 Log File

Below is an excerpt of the output file showing 20 steps, from 5060 to 5080, of the 200,000 in the log. In each step the arguments revealed by each agent, the accepted arguments in the action graph, the utility gained by each agent for each argument revealed that was accepted, the total pay gained by each agent and the updated policy of each agent. The policy of each agent is in the even numbered elements of the array, so for example in step 5060 the policies for agent 1 and agent 2, corresponding to P0 and P1 respectively, are (0.0636, 0.3804, 0.3090, 0.2467) and (0.0283, 0.0842, 0.3184, 0.5689).

```
    I::--- 5060 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 3; pay 0.00; 4.00;
I::P0[ 0.5843; 0.0636; 0.7147; 0.3804; 0.7601; 0.3090; 0.7935; 0.2467;]
```

```
I::P1[ 1.9262; 0.0283; 1.9062; 0.0842; 2.0437; 0.3184; 2.0556; 0.5689;]
I::--- 5061 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed C
D::  args revealed B
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 2; pay 0.00; 4.00;
I::P0[ 0.5815; 0.0636; 0.7076; 0.3804; 0.7540; 0.3090; 0.7915; 0.2468;]
I::P1[ 1.9280; 0.0283; 1.9121; 0.0841; 2.0632; 0.3185; 2.0731; 0.5689;]
I::--- 5062 ---
D::  The args revealed by Agent 0 are [A, C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed A
D::  args revealed C
D::  args revealed B
D::  The in Args are[A]
D::  Utility for a0 winning argument A is 1.0
I::  action 3; 2; pay 1.00; 0.00;
I::P0[ 0.5833; 0.0636; 0.7102; 0.3804; 0.7557; 0.3090; 0.7936; 0.2469;]
I::P1[ 1.9265; 0.0283; 1.9072; 0.0841; 2.0426; 0.3185; 2.0563; 0.5689;]
I::--- 5063 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed C
D::  args revealed B
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 2; pay 0.00; 4.00;
I::P0[ 0.5811; 0.0635; 0.7031; 0.3803; 0.7508; 0.3090; 0.7865; 0.2469;]
I::P1[ 1.9280; 0.0283; 1.9120; 0.0841; 2.0622; 0.3185; 2.0705; 0.5689;]
I::--- 5064 ---
D::  The args revealed by Agent 0 are [A, C]
D::  The args revealed by Agent 1 are [D]
D::  args revealed A
D::  args revealed C
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 3; 1; pay 1.00; 1.00;
I::P0[ 0.5825; 0.0635; 0.7058; 0.3803; 0.7523; 0.3091; 0.7886; 0.2470;]
I::P1[ 1.9274; 0.0283; 1.9029; 0.0841; 2.0526; 0.3185; 2.0635; 0.5689;]
I::--- 5065 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 3; pay 0.00; 4.00;
I::P0[ 0.5807; 0.0634; 0.6987; 0.3802; 0.7483; 0.3091; 0.7815; 0.2470;]
I::P1[ 1.9286; 0.0283; 1.9218; 0.0840; 2.0684; 0.3185; 2.0829; 0.5690;]
I::--- 5066 ---
D::  The args revealed by Agent 0 are [A, C]
```

```
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed A
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 3; 3; pay 1.00; 1.00;
I::P0[ 0.5819; 0.0634; 0.7014; 0.3802; 0.7495; 0.3091; 0.7837; 0.2471;]
I::P1[ 1.9281; 0.0283; 1.9143; 0.0840; 2.0606; 0.3185; 2.0720; 0.5690;]
I::--- 5067 ---
D::  The args revealed by Agent 0 are [A, C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed A
D::  args revealed C
D::  args revealed B
D::  The in Args are[A]
D::  Utility for a0 winning argument A is 1.0
I::  action 3; 2; pay 1.00; 0.00;
I::P0[ 0.5830; 0.0634; 0.7039; 0.3802; 0.7505; 0.3091; 0.7859; 0.2472;]
I::P1[ 1.9272; 0.0283; 1.9004; 0.0840; 2.0400; 0.3185; 2.0534; 0.5690;]
I::--- 5068 ---
D::  The args revealed by Agent 0 are [A]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed A
D::  args revealed B
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 2; 3; pay 1.00; 1.00;
I::P0[ 0.5839; 0.0633; 0.7060; 0.3801; 0.7530; 0.3091; 0.7878; 0.2472;]
I::P1[ 1.9268; 0.0283; 1.8945; 0.0840; 2.0306; 0.3185; 2.0428; 0.5690;]
I::--- 5069 ---
D::  The args revealed by Agent 0 are [A, C]
D::  The args revealed by Agent 1 are []
D::  args revealed A
D::  args revealed C
D::  The in Args are[A]
D::  Utility for a0 winning argument A is 1.0
I::  action 3; 0; pay 1.00; 0.00;
I::P0[ 0.5848; 0.0633; 0.7080; 0.3801; 0.7553; 0.3091; 0.7899; 0.2473;]
I::P1[ 1.9075; 0.0283; 1.8833; 0.0840; 2.0142; 0.3185; 2.0245; 0.5690;]
I::--- 5070 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 3; pay 0.00; 4.00;
I::P0[ 0.5837; 0.0633; 0.7009; 0.3800; 0.7491; 0.3091; 0.7828; 0.2474;]
I::P1[ 1.9264; 0.0283; 1.8945; 0.0839; 2.0287; 0.3186; 2.0442; 0.5690;]
I::--- 5071 ---
D::  The args revealed by Agent 0 are [C]
```

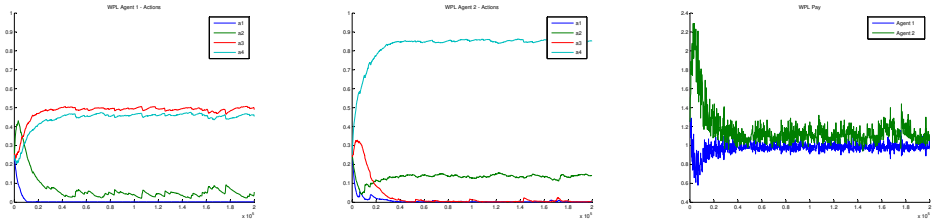
```
D::  The args revealed by Agent 1 are [B]
D::  args revealed C
D::  args revealed B
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 2; pay 0.00; 4.00;
I::P0[ 0.5827; 0.0632; 0.6939; 0.3800; 0.7437; 0.3091; 0.7765; 0.2474;]
I::P1[ 1.9432; 0.0283; 1.9046; 0.0839; 2.0484; 0.3186; 2.0618; 0.5691;]
I::--- 5072 ---
D::  The args revealed by Agent 0 are [A]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed A
D::  args revealed B
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 2; 3; pay 1.00; 1.00;
I::P0[ 0.5833; 0.0632; 0.6966; 0.3800; 0.7462; 0.3091; 0.7781; 0.2475;]
I::P1[ 1.9363; 0.0283; 1.9007; 0.0839; 2.0389; 0.3186; 2.0512; 0.5691;]
I::--- 5073 ---
D::  The args revealed by Agent 0 are [A]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed A
D::  args revealed B
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 2; 3; pay 1.00; 1.00;
I::P0[ 0.5839; 0.0632; 0.6991; 0.3799; 0.7488; 0.3092; 0.7795; 0.2476;]
I::P1[ 1.9302; 0.0283; 1.8972; 0.0839; 2.0305; 0.3186; 2.0407; 0.5691;]
I::--- 5074 ---
D::  The args revealed by Agent 0 are [A]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed A
D::  args revealed B
D::  args revealed D
D::  The in Args are[A, D]
D::  Utility for a0 winning argument A is 1.0
D::  Utility for a1 winning argument D is 1.0
I::  action 2; 3; pay 1.00; 1.00;
I::P0[ 0.5844; 0.0631; 0.7013; 0.3799; 0.7513; 0.3092; 0.7808; 0.2476;]
I::P1[ 1.9247; 0.0282; 1.8941; 0.0838; 2.0230; 0.3186; 2.0303; 0.5691;]
I::--- 5075 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 3; pay 0.00; 4.00;
I::P0[ 0.5838; 0.0631; 0.6943; 0.3799; 0.7445; 0.3092; 0.7767; 0.2477;]
I::P1[ 1.9357; 0.0282; 1.9007; 0.0838; 2.0360; 0.3186; 2.0500; 0.5691;]
I::--- 5076 ---
D::  The args revealed by Agent 0 are [A]
```

```
D::  The args revealed by Agent 1 are [B]
D::  args revealed A
D::  args revealed B
D::  The in Args are[A]
D::  Utility for a0 winning argument A is 1.0
I::  action 2; 2; pay 1.00; 0.00;
I::P0[ 0.5842; 0.0630; 0.6970; 0.3798; 0.7471; 0.3092; 0.7778; 0.2477;]
I::P1[ 1.9264; 0.0282; 1.8953; 0.0838; 2.0156; 0.3186; 2.0315; 0.5691;]
I::--- 5077 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B, D]
D::  args revealed C
D::  args revealed B
D::  args revealed D
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 3; pay 0.00; 4.00;
I::P0[ 0.5837; 0.0630; 0.6900; 0.3798; 0.7404; 0.3092; 0.7744; 0.2478;]
I::P1[ 1.9354; 0.0282; 1.9007; 0.0838; 2.0335; 0.3186; 2.0512; 0.5692;]
I::--- 5078 ---
D::  The args revealed by Agent 0 are [C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed C
D::  args revealed B
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 1; 2; pay 0.00; 4.00;
I::P0[ 0.5832; 0.0630; 0.6831; 0.3797; 0.7344; 0.3092; 0.7714; 0.2479;]
I::P1[ 1.9434; 0.0282; 1.9055; 0.0838; 2.0532; 0.3186; 2.0687; 0.5692;]
I::--- 5079 ---
D::  The args revealed by Agent 0 are []
D::  The args revealed by Agent 1 are [B]
D::  args revealed B
D::  The in Args are[B]
D::  Utility for a1 winning argument B is 4.0
I::  action 0; 2; pay 0.00; 4.00;
I::P0[ 0.5774; 0.0629; 0.6770; 0.3797; 0.7290; 0.3092; 0.7687; 0.2479;]
I::P1[ 1.9505; 0.0282; 1.9098; 0.0837; 2.0726; 0.3186; 2.0844; 0.5692;]
I::--- 5080 ---
D::  The args revealed by Agent 0 are [A, C]
D::  The args revealed by Agent 1 are [B]
D::  args revealed A
D::  args revealed C
D::  args revealed B
D::  The in Args are[A]
D::  Utility for a0 winning argument A is 1.0
I::  action 3; 2; pay 1.00; 0.00;
I::P0[ 0.5812; 0.0629; 0.6796; 0.3796; 0.7308; 0.3092; 0.7710; 0.2480;]
I::P1[ 1.9444; 0.0282; 1.9062; 0.0837; 2.0519; 0.3186; 2.0692; 0.5692;]
```

B.3.3 Sample Graphs

Figure B.2 shows the graphs that are plotted in Matlab using data extracted from the log file. The first two graphs are plots of the agents' policies against the step number, hence each graph has four plots corresponding to the probabilities of each of the four actions and shows the convergence, in this case, to roughly $(0, 0, \frac{1}{2}, \frac{1}{2})$ for agent 1 and $(0, \frac{1}{5}, 0, \frac{4}{5})$ for agent 2. The

third graph is a plot of the pay of each agent against the step number which roughly converges to (1,1).

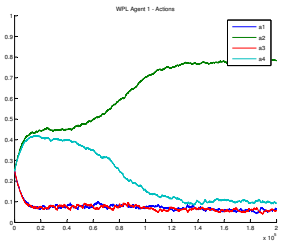


(a) Agent 1 (b) Agent 2 (c) Pay

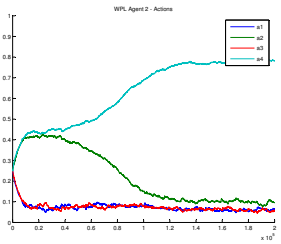
Figure B.2: Sample Graphs for resulting 2-player 4-action Game

Appendix C

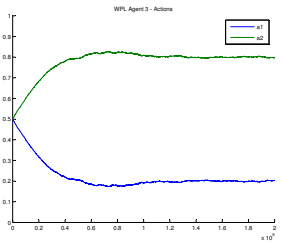
Additional Results



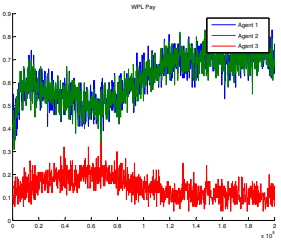
(a) Agent 1



(b) Agent 2

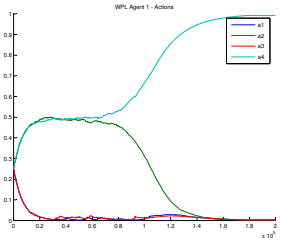


(c) Agent 3

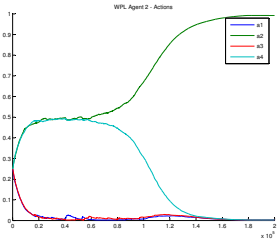


(d) Pay

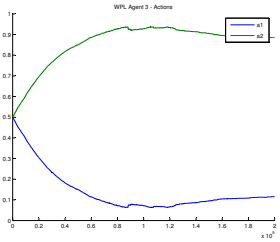
Figure C.1: Game 5v1: WPL α=0.1 δ=0.002 ϵ=0.01



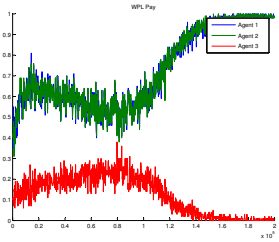
(a) Agent 1



(b) Agent 2

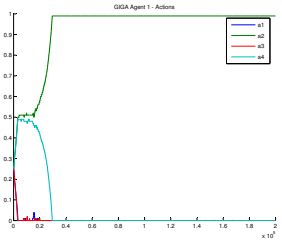


(c) Agent 3

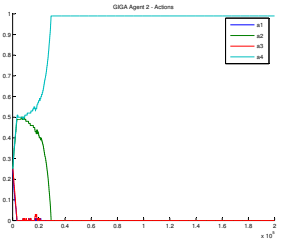


(d) Pay

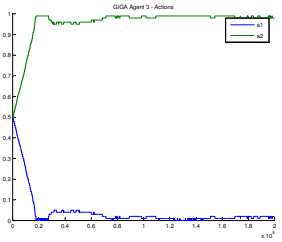
Figure C.2: Game 5v1: WPL α=0.01 δ=0.002 ϵ=0.01



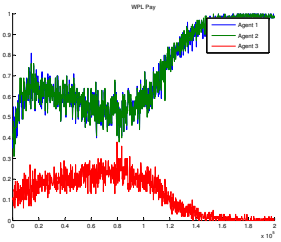
(a) Agent 1



(b) Agent 2

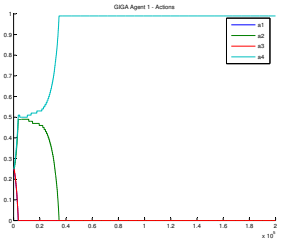


(c) Agent 3

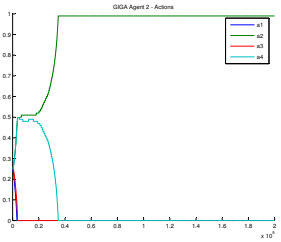


(d) Pay

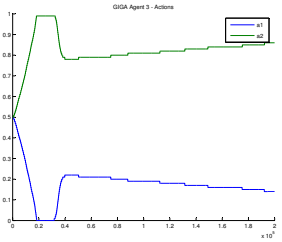
Figure C.3: Game 5v1: GIGA α=0.01 δ=0.002 ϵ=0.01



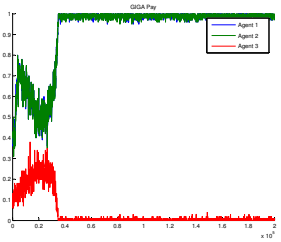
(a) Agent 1



(b) Agent 2



(c) Agent 3



(d) Pay

Figure C.4: Game 5v1: GIGA α=0.001 δ=0.002 ϵ=0.01

Bibliography

[1] Sherief Abdallah and Victor Lesser. Learning the Task Allocation Game. In *5th International Joint Conference on Autonomous Agents & Multi Agent Systems, AAMAS'2006, Hakodate, Hokkaido, Japan*, 2006.

[2] Sherief Abdallah and Victor Lesser. A Multiagent Reinforcement Learning Algorithm with Non-linear Dynamics. In *Journal of Artificial Intelligence Research*, volume 33, pages 521–549, 2008.

[3] Trevor J. M. Bench-Capon and Paul E. Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171(10–15):619–641, 2007.

[4] Philippe Besnard and Anthony Hunter. *Elements of Argumentation*. MIT Press, 2008.

[5] Michael Bowling. Convergence and No-regret in Multiagent Learning. In *Annual Conference on Advances in Neural Information Processing Systems*, pages 209–216, 2005.

[6] Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

[7] Martin Caminada. *A Gentle Introduction to Argumentation Semantics*, 2007.

[8] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 746–752. AAAI Press, 1998.

[9] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[10] John Fox, David Glasspool, Dan Grecu, Sanjay Modgil, Matthew South, and Vivek Patkar. Argumentation-based inference and decision making–a medical perspective. *IEEE Intelligent Systems*, 22(6):34–41, 2007.

[11] Ronald A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge MA, USA, 1960.

[12] Anthony Hunter. Real arguments are approximate arguments. In *Proceedings of the 22nd Conference on Artificial Intelligence*, pages 66–71. AAAI Press, 2007.

[13] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, New York NY, USA, 1995.

[14] Irene Mazzotta, Fiorella de Rosis, and Valeria Carofiglio. Portia: A user-adapted persuasion system in the healthy-eating domain. *IEEE Intelligent Systems*, 22(6):42–51, 2007.

[15] Simon Parsons, Carles Sierra, and Nick R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8:261–292, 1998.

[16] Henry Prakken and Giovanni Sartor. Argument-based extended logic programming with defeasible priorities. In Pierre-Yves Schobbens, editor, *Working Notes of 3rd ModelAge Workshop: Formal Models of Agents*, Sesimbra, Portugal, 1996.

[17] Henry Prakken and Gerard Vreeswijk. Logics for defeasible argumentation. In D.Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 4. Kluwer Academic Publishers, Dordrecht etc, 2nd edition, 2002.

[18] Iyad Rahwan and Kate Larson. Mechanism Design for Abstract Argumentation. In L. Padgham, D. Parkes, J. Mueller, and S. Parsons, editors, *7th International Joint Conference on Autonomous Agents & Multi Agent Systems, AAMAS'2008, Estoril, Portugal*, 2008.

[19] Iyad Rahwan and Kate Larson. Pareto Optimality in Abstract Argumentation. In *Proceedings of the 23rd Conference on Artificial Intelligence*, pages 150–155, California, USA, 2008. AAAI Press.

[20] Iyad Rahwan and Kate Larson. Argumentation and game theory. In Iyad Rahwan and Guillermo R. Simari, editors, *Argumentation in Artificial Intelligence*. Springer, 2009.

[21] Iyad Rahwan and Peter McBurney. Guest editors' introduction: Argumentation technology. *IEEE Intelligent Systems*, 22(6):21–23, 2007.

[22] Eric Rasmusen. *Games and Information: An Introduction to Game Theory*. Blackwell, 4th edition, 2006.

[23] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, New York NY, USA, 2009.

[24] Satinder Singh, Michael Kearns, and Yishay Mansour. Nash Convergence of Gradient Dynamics in General-sum Games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 541–548. Morgan, 2000.

[25] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning*. MIT Press, 1998.

[26] Paolo Torroni, Marco Gavanelli, and Federico Chesani. Argumentation in the semantic web. *IEEE Intelligent Systems*, 22(6):66–74, 2007.

[27] Frans H. van Eemeren, Rob F. Grootendorst, and Francisca S. Henkemans, editors. *Fundamentals of Argumentation Theory: A Handbook of Historical Backgrounds and Contemporary Applications*. Lawrence Erlbaum Associates, Hillsdale NJ, USA, 1996.

[28] Christopher J.C.H Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge, England, 1989.

[29] Martin Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *Proceedings of the International Conference on Machine Learning*, pages 928–936, 2003.