UNIVERSITY OF SOUTHAMPTON

Practical Strategies for Agent-Based Negotiation in Complex Environments

by

Colin Richard Williams

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Physical and Applied Sciences
Electronics and Computer Science

December 2012

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF PHYSICAL AND APPLIED SCIENCES
Electronics and Computer Science

Doctor of Philosophy

Practical Strategies for Agent-Based Negotiation in Complex Environments

by Colin Richard Williams

Agent-based negotiation, whereby the negotiation is automated by software programs, can be applied to many different negotiation situations, including negotiations between friends, businesses or countries. A key benefit of agent-based negotiation over human negotiation is that it can be used to negotiate effectively in complex negotiation environments, which consist of multiple negotiation issues, time constraints, and multiple unknown opponents. While automated negotiation has been an active area of research in the past twenty years, existing work has a number of limitations. Specifically, most of the existing literature has considered time constraints in terms of the number of rounds of negotiation that take place. In contrast, in this work we consider time constraints which are based on the amount of time that has elapsed. This requires a different approach, since the time spent computing the next action has an effect on the utility of the outcome, whereas the actual number of offers exchanged does not. In addition to these time constraints, in the complex negotiation environments which we consider, there are multiple negotiation issues, and we assume that the opponents' preferences over these issues and the behaviour of those opponents are unknown. Finally, in our environment there can be concurrent negotiations between many participants.

Against this background, in this thesis we present the design of a range of practical negotiation strategies, the most advanced of which uses Gaussian process regression to coordinate its concession against its various opponents, whilst considering the behaviour of those opponents and the time constraints. In more detail, the strategy uses observations of the offers made by each opponent to predict the future concession of that opponent. By considering the discounting factor, it predicts the future time which maximises the utility of the offers, and we then use this in setting our rate of concession.

Furthermore, we evaluate the negotiation agents that we have developed, which use our strategies, and show that, particularly in the more challenging scenarios, our most advanced strategy outperforms other state-of-the-art agents from the Automated Negotiating Agent Competition, which provides an international benchmark for this work. In more detail, our results show that, in one-to-one negotiation, in the highly discounted scenarios, our agent reaches outcomes which, on average, are 2.3% higher than those of the next best agent. Furthermore, using empirical game theoretic analysis we show the robustness of our strategy in a variety of tournament settings. This analysis shows that, in the highly discounted scenarios, no agent can benefit by choosing a different strategy (taken from the top four strategies in that setting) than ours. Finally, in the many-to-many negotiations, we show how our strategy is particularly effective in highly competitive scenarios, where it outperforms the state-of-the-art many-to-many negotiation strategy by up to 45%.

Contents

List of Tables

List of Figures

Declaration of Authorship

I, Colin Richard Williams

declare that the thesis entitled

Practical Strategies for Agent-Based Negotiation in Complex Environments

and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as:
  - Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2010) IAMhaggler: A Negotiation Agent for Complex Environments. In: *New Trends in Agent-based Complex Automated Negotiations, Series of Studies in Computational Intelligence, Springer-Verlag* **383** 151-158.
  - Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2011) Using Gaussian Processes to Optimise Concession in Complex Negotiations against Unknown Opponents. In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, AAAI Press* **1** 432-438.
  - Baarslag, T., Fujita, K., Gerding, E. H., Hindriks, K., Ito, T., Jennings, N. R., Jonker, C., Kraus, S., Lin, R., Robu, V. and Williams, C. R. (2012) Evaluating Practical Negotiating Agents: Results and Analysis of the 2011 International Competition. In: *Artificial Intelligence* (sub).
  - Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2012) IAMhaggler2011: A Gaussian Process Regression based Negotiation Agent. In: *Complex Automated Negotiations: Theories, Models, and Software Competitions, Series of Studies in Computational Intelligence, Springer-Verlag* **435** 209-212.
  - Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2012) Negotiating Concurrently with Unknown Opponents in Complex, Real-Time Domains. In: *20th European Conference on Artificial Intelligence* pp. 834-839.

Signed: ....

Date: ....

Acknowledgements

First and foremost, I thank my supervisors, Prof. Nick Jennings and Dr. Enrico Gerding who, along with Dr. Valentin Robu have formed a superb supervisory team. Together, they have provided excellent support and guidance that has enabled me to complete this PhD thesis and the publications associated with it. I also thank the ALADDIN project (funded jointly by BAE Systems and EPSRC, grant number EP/C548051/1) for funding the completion of this PhD, and I acknowledge the use of the IRIDIS High Performance Computing Facility, and associated support services, in the completion of this work.

Secondly, I thank the members of the *Interactive Intelligence Group* at Delft University of Technology, specifically Prof. Catholijn Jonker, Dr. Koen Hindriks, Dr. Dmytro Tykhonov and Tim Baarslag. The GENIUS negotiation platform which they developed has been an important tool which I have used to assist my research, and it has been a pleasure to collaborate with this group from the early stages of my PhD. In addition, I thank the co-authors of the journal paper I was recently involved in writing: Prof. Takayuki Ito, Prof. Sarit Kraus, Dr. Katsuhide Fujita and Dr. Raz Lin (in addition to those from Southampton and Delft mentioned above).

Thirdly, I thank my colleagues in the *Agents, Interaction and Complexity* research group, particularly Muddasser Alam, Dr. Kathryn Macarthur, Oliver Parson, Oleksandr Pryymak, Harry Rose, Dr. Sebastian Stein, Dr. Ruben Stranders, Dr. Long Tran-Thanh and Matteo Venanzi. It has been a pleasure to work and socialise with such a great team, both on a day-to-day basis and at the conferences we have attended together.

Fourthly, I thank the many friends who have helped to make the past three years extremely enjoyable. In particular, I thank my close friends Kewalin Angkananon, Norhidayah Azman, Marcus Cobden, James Morse, Iman Naja, Dr. David Newman, Jarutas Pattanaphanchai, Betty Purwandari and Dr. David Tarrant.

Finally, I thank my family for their love and support, both before and during my PhD studies.

Nomenclature

i	issue
I	set of issues
n	number of issues
p	player
q	opponent
P	set of players
Ξ	domain
Ξ_p	preference profile for player p
o	offer
$o_{p,r}$	offer made by player p at round r
$o_{p,t}$	offer made by player p at time t
O	set of possible outcomes
$\|O\|$	size of outcome space O
$C(O)$	competitiveness of outcome space O
α	agreement
t	time
t_α	time at which agreement α was made
t_c	current time
r	round number
δ	discounting factor
δ_p	discounting factor for player p
c	bargaining cost
c_p	bargaining cost for player p
β	concession rate parameter
$\rho(t)$	decommitment penalty at time t
ρ_0	decommitment penalty at time 0
$\rho_{\max}$	decommitment penalty at the negotiation deadline
$U_p(o)$	utility to player p of offer o
$U_{p,\text{opp}}(t)$	utility to player p of the opponent's offers at time t
$U_{p,i}()$	utility function to player p for issue i

$w_{p,i}$	weight to player p of issue i
v_i	value for issue i
V_i	set of values that issue i can take
$\langle v_1, v_2 \rangle$	offer consisting of value v_1 in issue 1 and v_2 in issue 2
$D_p()$	discounting function for player p
U_0	utility at time 0
$U_{\min}$	reservation utility
$u_{\overline{\alpha}}$	utility of conflict
$S_p()$	spitefulness function for player p
s	spitefulness coefficient

Chapter 1

Introduction

Negotiation is a process in which a range of negotiating parties, with different desires, aim to reach agreement on a common set of issues. It is a task that has many practical applications. For example, a group of people may negotiate in an attempt to choose a restaurant to eat at. A client and supplier may negotiate in the sale of goods or services and negotiations also take place when forming employment contracts. Negotiations may even take place at the global scale, in the case of diplomatic negotiations between countries. However, despite the frequency with which humans take part in some form of negotiation, it is not an easy task for them to complete. It is often a time consuming process, and it is difficult for humans to find efficient agreements (Bosse and Jonker, 2005), especially when they have little negotiation experience (Raiffa, 1982; Lin and Kraus, 2010). Here, an efficient agreement is one where there is no other potential agreement that would increase the benefit to one party without decreasing the benefit of the other party. Therefore, there is considerable interest in automating the negotiation process, by using software agents to assist in the negotiation. In this context, an agent is a piece of software that acts on behalf of one of the negotiating parties and is able to interact autonomously with other agents (or humans) in order to reach an agreement (Wooldridge and Jennings, 1995). To this end, we provide the following motivating scenario:

It is Thursday afternoon, and Bob is starting a new job on Monday. His job involves a lot of travel, so he urgently needs to buy a car and is keen that the agreement to buy this car should be made as soon as possible. He informs his automated buying agent that he needs a small car which costs approximately £6,000. He is keen to limit his impact on the environment, but drives long distances, so he desires a battery-powered car with a high capacity battery. The colour of the car is not very important to Bob, but he prefers dark coloured cars, with his favourite colour being blue. Since he will be using the car for *work purposes, it is important that the car is in a good condition. The buying agent is notified of these additional preferences, and also of the time constraints for completing the deal.*

The buying agent then contacts the selling agents of all of the local dealers, in order to negotiate with them. As he lives in a large city, there are around 20 different dealers, all competing for Bob's business. During interactions with the selling agents, hundreds of different offers may be made. In addition, Bob's friend has offered his car for sale, though it's a rather inefficient car that Bob isn't too keen on. After a short while, Bob's agent agrees to buy a small blue car from Alpha Cars, at a price of £4,950. The car is in a good condition, but its battery capacity isn't great. This is the best deal that the agent has found, and negotiations don't seem to be progressing any further. Bob will be very pleased with this car at this price, and therefore the agent pays a non-refundable deposit of £200 to secure the deal.

Just minutes later, the agent from Beta Motors makes a slightly better offer: a car which is also in good condition, but this time in yellow and with a better battery capacity, at a price of £5,050. The buying agent knows that Bob will prefer this car over the one that he has placed a deposit on, but decides that it isn't worth losing the deposit for.

A new agent, representing Delta Vehicles then enters the market, and Bob starts to negotiate with this new agent, whilst continuing negotiations with the other dealers' agents. Throughout the negotiations, other buying agents, each with their own requirements enter and leave the market.

Following further negotiations, the Beta Motors agent proposes another car, similar to the previous one, but this time in black, and at just £5,000. Even though the deposit on the first car will be lost if the agent decides to change to this offer, the buying agent makes the decision, on behalf of Bob, that it is worth it. The price difference of £50 plus the £200 deposit that has already been paid to Alpha Cars, along with a slightly less preferred colour is far outweighed by the value that Bob places on securing a car with such a good battery capacity.

The buying agent therefore reaches an agreement to purchase the black car from Beta Motors. The deal is completed well within the time limits imposed by Bob's situation, and both parties are confident that they have got a good deal.

In addition to the retail application that we provide in our scenario, automated negotiation can also be applied to areas as diverse as electricity markets (Brazier et al., 1998), transportation scheduling (Fischer et al., 1995), and the trading of financial derivatives (Bichler, 2000).

In most negotiations similar to the one above (but without the use of automated agents), human negotiators may expect to reach an agreement following a relatively small number of interactions with their opponent. If the negotiation appears to be taking too long, it is likely that one of the participants will simply walk away. Each interaction would be likely to take many seconds or even minutes (Bosse and Jonker, 2005). In addition, it is unlikely that a human negotiator could simultaneously negotiate with so many opponents.

To overcome some of the limitations of human negotiation, we can use software agents. These autonomous agents are usually self-interested, and their aim is to maximise the value of the agreement from the point of view of the party that they represent. The value of an agreement for a given participant is known as its utility. For any agreement, this utility can be calculated according to the agent's utility function, which is based on the preferences of the participant. In an automated negotiation, it is possible for the agents to exchange tens of thousands of offers with each other before reaching an agreement. In addition, it is possible for them to use more complicated algorithms than a human would use, since the agents have less restrictive memory and computational constraints (although, to remain practical, the agents must not require unlimited memory or computational power).

For example, in a simple negotiation between two individuals, there may only be a single issue that they are trying to agree the value of (such as a price). In a more complex setting such as the one in our scenario, there are a range of different issues that are being negotiated over (the price, the battery capacity, the condition and the colour). This makes the negotiation much more difficult, as the seller is unlikely to know which features the buyer is most interested in. Similarly, the buyer will be unlikely to know which type of car the dealer would prefer to sell. By identifying the issues that are important to each party, it is possible to reach a better agreement. Some of the issues, such as price, have an ordering which is common to all negotiating parties. In contrast, other issues, such as colour, do not have a known, common order. For example, one party may prefer bright colours, whereas another may prefer warm colours. Although the agent's primary goal is to maximise the value of its agreements, in order to maximise the chance of such agreements occurring, it should attempt to make offers that are efficient. Such offers increase the probability of high value agreements being reached, since, of all of the possible outcomes with a given value to one party, they are more likely to be accepted by the other negotiating party. For example, in the car sales domain, the dealer may consider the colour of the car to be of little importance compared to the price, whereas for the buyer, the colour is much more important. Therefore, when choosing between two cars, one in a colour that the buyer dislikes, and the other in the buyer's preferred colour, but at a slightly higher price, the dealer may well be better off offering the more expensive car. As a result, the dealer increases their utility (by increasing the price) and at the same time, the buyer's utility is also increased (as the car is in the colour they prefer).

In addition to uncertainty regarding the preferences of the other party, each party is also uncertain about the behaviour of the other. The dealer may be keen to make the sale to one particular buyer, and will therefore be willing to offer a deal at a fairly low utility. On the other hand, the dealer may have many other potential buyers, and is therefore keen to reach an agreement with one of them, at a high value. Furthermore, in order to find a suitable agreement, the seller may make offers in decreasing order of preference, or he may simply make a variety of offers which he considers to be acceptable, but in no particular order. The same can be said about the behaviour of the buyer.

As seen in our scenario, the negotiation is further complicated in a realistic environment where there are more participants than just a single dealer and a single buyer. In such an environment, each buyer is able to negotiate with a number of dealers, and each dealer may have a number of customers. All of the parties can negotiate concurrently, but for humans this would be highly challenging, as they would need to simultaneously manage negotiations with a range of opponents. The dealer may find that, for a particular car, one of the buyers (A) is willing to pay more than another buyer (B), and it would therefore be wasteful to reach an agreement to sell this particular car to B.

Against this background, in this work, we consider an environment to be complex if it contains at least two of the following features:

- The negotiation domain contains many issues, resulting in a **large outcome space**.
- Each agent has **uncertainty** about many aspects of the other participants. This includes uncertainty about their utility functions, how concessive they will be, and even whether or not their behaviour will be rational.
- The environment imposes **time constraints** which affect the value of any agreement that is reached. Moreover, these time constraints may be based on the amount of real time that has elapsed.
- The environment allows **concurrent negotiation** between more than a single pair of agents.

In more detail, in Chapters 3 and 4 we focus on negotiation scenarios which contain uncertainty and time constraints, and in many cases, large outcome spaces. In Chapters 5 and 6, we consider negotiation scenarios which contain all of the above features.

In the remainder of this chapter, we provide a summary of the automated negotiation challenges that exist in the scenario that we introduced in this section. We then detail the objectives of our work before briefly describing our contributions. We conclude this chapter with an outline of the rest of this thesis.

1.1 Research Challenges of Complex Environments

In complex automated negotiations there are a number of challenges that need to be solved. Specifically, we consider the following to be the key challenges which we aim to address in our work:

1.1.1 Domains with a Large Outcome Space

Domains in which there are a large number of possible outcomes that could be agreed upon are said to have a large outcome space. In negotiations over such domains, finding a package that is acceptable to both parties becomes more of a challenge than in a smaller domain, where it may be possible to propose a large proportion of the packages during the negotiation. Domains which contain multiple issues, such as the one in our scenario, contain a possible outcome for each combination that can be formed from the values of each issue. Consequently, such domains may have a large outcome space. In the car sales domain, the issues are the price, the colour, the battery capacity and the car's condition. If we assume that there are 10 possible colours, that price is discretised into multiples of £50 in the range £4,500 to £5,500 (21 possible prices), and that there are 5 possible values for both the battery capacity and the car's condition, we obtain a domain with $10 * 21 * 5 * 5 = 5250$ possible outcomes.

1.1.2 Uncertainty about the Opponents

Due to the competitive nature of automated negotiation, the negotiating parties will typically be unwilling to disclose information about their preferences. Therefore, it is often required for an agreement to be reached in an environment where there is uncertainty about the preferences of the other parties and where there may be uncertainty about any time constraints (see Section 1.1.3 for more details of the time constraints).

In more detail, in negotiations with multiple issues, there is the opportunity to reach more efficient agreements by finding mutually agreeable trade-offs between issues, in order to reach 'win-win' agreements. However, in a negotiation where there is uncertainty about the preferences of the opponent, it is impossible for either party to know which agreements are more efficient. Therefore, a common approach is to approximate the opponent's preferences and constraints, thereby allowing the agent to estimate which agreements are more efficient. In our scenario, there are multiple issues, and the car dealers' agents do not know how much the customer values each issue (such as the battery capacity or price of the car). Now, if the dealers' agents could estimate the importance of each issue to the buyer, they would consider the battery capacity to be more significant than the car's price. This would allow them to make offers with a better battery capacity but at a higher price, therefore increasing the value of the agreement for both parties.

Even when considering just one of the negotiation issues, there may or may not be a known, common ordering over the values of that issue. In the car sales domain, an example of an issue with a common ordering is the car's condition, for which both agents consider the values to have the ordering: 'excellent', 'very good', 'average', 'poor', 'bad'. Each agent's utility function for the issue assigns similar utilities to nearby values. Therefore, both agents are aware that a car which is in 'excellent' condition will have a similar utility to one in 'very good' condition but quite different to one in a 'bad' condition. In contrast, colour is an example of an issue without a common ordering. That is, one party may consider red and pink to be similar to each other, therefore having similar utilities, whilst the other party may consider those two colours to be very different to each other, and therefore have highly different utilities.

Furthermore, since any approximation of the opponent's preferences only provides an estimate of the true preferences, it is important that an agent does not rely entirely on that estimate, as there is a chance that it may be incorrect. If the agent bases its offers too heavily on a poor estimate of the opponent's true preferences, it is possible that many good potential offers would not be considered, therefore leading to a less efficient agreement. Since there are many real-world domains in which there is uncertainty about the opponent's preferences and behaviour, the problem of dealing with uncertainty has become a popular topic of research in automated negotiation, and we will review work related to it in Chapter 2.

1.1.3 Time Constraints

When performing negotiation, it is often important that the negotiation is completed within a limited amount of time. For example, in the car sales scenario, the buyer needs to complete his purchase before a specific date. In order to encourage agreements to be reached in a timely manner, time constraints are used.

There are two key approaches to time constraints that are generally considered. These different time constraints are known as a *discounting factor* and a *deadline*. The former is a cost that is applied based on the duration of the negotiation. It has the effect that any particular agreement that is reached at a given time is of higher value than that same agreement at a later time. This causes agents that are slow to reach agreement to be punished, as the value of the agreement will be reduced due to the time delay. The deadline is a point in time before which an agreement needs to be made for it to be of any value. A deadline imposes a limit on the maximum duration of negotiation, as there is no value in an agent continuing to negotiate beyond the deadline.

In our work, we consider both of these types of time constraint, as both are relevant to our scenario. The discounting factor represents the value that the buyer (or the dealer) places on reaching an agreement sooner rather than later. The deadline exists as the buyer needs to buy the car before starting his new job, and therefore their agreement must be made before this time.

Due to the existence of these time constraints, it is necessary for the agent to negotiate at an appropriate rate. That is, it is important that the agent does not 'give in' too quickly to its opponent, unless the opponent is willing to make a similar compromise. However, it is also important that the agent does not take an approach that is too tough, as this could delay the negotiation unnecessarily, causing the value of an agreement to be decreased. Furthermore, in domains with a deadline, an increased delay increases the likelihood that the agents will fail to reach an agreement. It is therefore a challenge to develop a negotiation strategy that will maximise the agent's gains.

In order to simplify the negotiation protocol and strategy, time constraints can be applied based on the discrete number of offers that are made during the negotiation (regardless of the amount of elapsed time), and this approximation has been widely researched (Lai et al., 2006; Yoshikawa et al., 2008; Yasumura et al., 2009; Fatima et al., 2001, 2002, 2004, 2006, 2007). Alternatively, the constraints can be applied based on the amount of physical elapsed time rather than the number of interactions (Soh and Tsatsoulis, 2005). We consider that constraints based on elapsed time are more appropriate for automated negotiation in the real world, as the cost of each interaction is likely to be minimal compared to the cost of delaying the time of agreement.

Now, using constraints based on the amount of elapsed time makes it more difficult to predict the utility of an offer that an agent might make in the future (other than during the current step). This is due to the fact that the time at which any future offer will be made by an agent depends on the time that its opponent spends in making its offers. For the same reason, it is also impossible to know how many interactions will occur before the deadline is reached. Therefore, another party is able to influence an agent's utility by delaying the negotiation.

1.1.4 Concurrent Negotiations

In situations in which there are more than two parties, which is likely to be the case in many real-world scenarios, such as in an extension of the scenario we provided where there could be multiple customers negotiating concurrently with a number of car dealers, there are significant additional challenges.

In order to reach efficient agreements in concurrent negotiations, it is necessary to consider not only whether the outcome is efficient for a specific pair of agents, but also whether a more efficient agreement can be formed with a different opponent. The most efficient agreements that can be reached are those which are between pairs of agents which have similar preferences.

Furthermore, in concurrent negotiations, since each opponent may be negotiating with other parties, it is possible that an opponent will leave the negotiation before the deadline, since it has reached an agreement with one of the other parties. Therefore, the negotiating agent cannot be sure how long an opponent will remain in the negotiation, and the number of opponents can vary throughout the negotiation.

Additionally, the environment may allow either party to breach an agreement. This decommitment represents a further challenge that is associated with concurrent negotiation, since it requires the agent to consider when it is appropriate to breach an agreement. It has been shown that, by allowing decommitment, more efficient agreements can be reached, and therefore an environment which allows decommitment is desirable (Andersson and Sandholm, 2001; Sandholm and Lesser, 2001; 't Hoen and Poutré, 2004). However, it is also important that a penalty is applied when an agent breaches an agreement, otherwise unnecessary decommitment can occur, which damages the efficiency of the agents (Ponka, 2009). This further adds to the challenge, since, depending on the negotiation protocol, it may be necessary for the agent to set the penalties as part of the negotiation process.

1.2 Research Requirements

Based on the challenges given in Section 1.1, the objectives of this research are to design an autonomous agent that:

1. works in a **decentralised** manner: The agent should not require a central entity, which acts as a mediator, to assist with the negotiation. Each agent should communicate directly with the other agents it is negotiating with.

2. is able to reach **efficient** agreements: Any agreement that is reached should be efficient, in that it should ensure that little or no utility is wasted. In more detail, an efficient agreement is one where it is not possible for one agent to increase its utility without reducing the utility of any opponents.

3. is able to negotiate against **unknown opponents**: The agent should be designed to perform well without knowledge of the preferences or behaviour of any other party. It should be able to negotiate in a one-off negotiation with the opponent, without needing to learn the opponent's behaviour over a series of negotiations. Furthermore, the agent should perform well if the opponent is using a similar (or the same) behaviour to itself.

4. is able to negotiate over **multiple issues**: The agent should be able to negotiate in domains with many issues, which may have hundreds of thousands of potential outcomes.

5. is able to negotiate over discrete issues with an **unknown ordering**: The agent should not require the issues to have a known, common ordering.

6. supports **real-time constraints**: Any time constraints will be based on the amount of time taken to produce an offer rather than the number of negotiation steps. In automated negotiation, the amount of time that has elapsed is more important than the number of steps, and therefore, the number of steps is not considered in this work.

7. uses a **computationally tractable** approach: The agent needs to use an approach which is computationally tractable, in order to be able to propose each offer within a limited finite time, in the order of a few seconds, using finite computational resources.

8. is able to effectively **coordinate** multiple concurrent negotiations: The agent should be designed to coordinate multiple concurrent negotiations with a range of opponents in order to reach effective outcomes.

9. takes advantage of **decommitment** by optimising when to decommit: The agent should decommit from agreements when it is beneficial for it to do so in order to improve its utility, considering the benefit of the new agreement over the decommitment penalty.

1.3 Research Contributions

Against the requirements set out in Section 1.2, we have developed a range of negotiating strategies and have implemented negotiating agents which use these strategies.

Specifically, our contributions to the state-of-the-art are as follows:

- We develop a novel strategy, which uses both a Gaussian process prediction and the certainty of that prediction, to calculate the concession an agent should make over time. This strategy is able to negotiate directly (Requirement 1) with an unknown opponent (Requirement 3) and uses a principled approach, by firstly predicting the opponent's future behaviour and then adapting to the agent's offers in order to maximise the expected utility of agreement. Furthermore, the proposed strategy is the first practical (Requirement 7) concession strategy which has been designed to deal with real-time constraints (Requirement 6) in multi-issue negotiation (Requirement 4). It reaches efficient agreements (Requirement 2) in scenarios where the issues have an unknown ordering (Requirement 5).
- We extend our strategy to support the coordination of multiple, concurrent negotiations (Requirement 8) in which each participant aims to reach a single agreement, and decommitment of agreements is allowed, through payment of a penalty (Requirement 9). Our strategy coordinates the concession rates for each opponent by considering the observed behaviour of all of the opponents. We show that it outperforms an existing state-of-the-art strategy for coordinating multiple negotiation threads (Nguyen and Jennings, 2005), in a range of scenarios.
- We empirically evaluate our strategies against those of a number of other state-of-the-art agents, developed for the international Automated Negotiating Agents Competition. We thereby show that, in a negotiation tournament consisting of a range of scenarios with significant discounting factors (Requirement 6), if all agents use our strategy, there is no incentive for any single agent to deviate to a different strategy used by one of the set of other state-of-the-art agents we consider.

The work that has been completed as part of this thesis has resulted in the development of a number of different negotiating strategies, along with the implementation of negotiation agents which use these strategies.

The first such agent, now known as *IAMhaggler2010*, finished in fourth place in the first Automated Negotiating Agents Competition (ANAC 2010) which was held at the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010). A paper describing this strategy was published as:

- Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2010) IAMhaggler: A Negotiation Agent for Complex Environments. In: *New Trends in Agent-based Complex Automated Negotiations, Series of Studies in Computational Intelligence, Springer-Verlag* **383** 151-158.

Section 3.2.1 and Appendix B are based on the above paper.

We have also created a further negotiating agent, known as *IAMhaggler2011*, which finished in third place in the second Automated Negotiating Agents Competition (ANAC 2011) which was held at the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011). Papers describing this strategy were published as:

- Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2012) IAMhaggler2011: A Gaussian Process Regression based Negotiation Agent. In: *Complex Automated Negotiations: Theories, Models, and Software Competitions, Series of Studies in Computational Intelligence, Springer-Verlag* **435** 209-212.
- Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2011) Using Gaussian Processes to Optimise Concession in Complex Negotiations against Unknown Opponents. In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, AAAI Press* **1** 432-438.

Sections 3.2.2 and 4.7 are based on the above papers.

Furthermore, we have created a negotiation strategy for many-to-many negotiation. The agent which uses this strategy is referred to as *IAMconcurrentHaggler*, and was presented in:

- Williams, C. R., Robu, V., Gerding, E. H. and Jennings, N. R. (2012) Negotiating Concurrently with Unknown Opponents in Complex, Real-Time Domains. In: *20th European Conference on Artificial Intelligence* pp. 834-839.

Chapter 6 is based on the above paper.

Finally, parts of our empirical evaluation have been published in:

- Baarslag, T., Fujita, K., Gerding, E. H., Hindriks, K., Ito, T., Jennings, N. R., Jonker, C., Kraus, S., Lin, R., Robu, V. and Williams, C. R. (2012) Evaluating Practical Negotiating Agents: Results and Analysis of the 2011 International Competition. In: *Artificial Intelligence* (in press).

The work in Section 4.7 is an updated version of part of the above paper.

Although parts of the thesis are based on the above papers, the work presented in the evaluation chapters (Chapters 4 and 6) has been updated by testing the performance of our strategies against other recently developed state-of-the-art strategies for complex real-time negotiation environments.

1.4 Report Outline

The remainder of the thesis is organised as follows:

- In Chapter 2, we review the related literature, considering both single-issue and multi-issue negotiation. Additionally, we consider the literature relating to concurrent negotiation.
- In Chapter 3, we present the theory behind the strategy that we have developed for one-to-one negotiation. This is split into two key parts, our strategy for negotiating efficiently under uncertainty and our concession strategy.
- In Chapter 4, we evaluate our one-to-one negotiation strategy, using the domains and strategies developed for the Automated Negotiating Agents Competition. We show its performance in both a tournament and in self-play.
- In Chapter 5, we extend our one-to-one negotiation to develop a strategy for many-to-many negotiations.
- In Chapter 6, we evaluate our many-to-many negotiation strategy, comparing it with an existing state-of-the-art strategy for many-to-many negotiation, using the strategies developed for the Automated Negotiating Agents Competition as opponents.
- In Chapter 7, we conclude, and present our ideas for future work.

Chapter 2

Background and Related Work

In this chapter we discuss previous work on automated negotiation. The chapter begins with definitions of the basic terminology used in this field (see Section 2.1). In the subsequent section (2.2), we consider the evaluation criteria and methodologies that are used to evaluate solutions to negotiation problems. The chapter then reviews different negotiation strategies, considering game theoretic approaches, heuristic approaches and argumentation based negotiation (see Section 2.3). The chapter concludes with a review of negotiation techniques when there are more than two parties (see Section 2.4). The chapter is summarised in Section 2.5.

2.1 Terminology

2.1.1 Bi-Lateral Negotiation

Bi-lateral negotiation (bargaining) is a form of interaction in which two self-interested parties aim to reach a mutual agreement in order to fulfil their goal. The two parties negotiate over a set of issues (see Section 2.1.2), with their aim being to maximise their utility (see Section 2.1.3) by reaching an agreement that suits their preferences. Furthermore, there may be time constraints imposed in order to encourage an agreement to be reached quickly (see Section 2.1.4). In many negotiations, the negotiation parties may have incomplete information about various aspects of the opponent, including their preferences (see Section 2.1.6). During the negotiation, each party must use a common protocol (see Section 2.1.7), but they are able to choose their own strategy (see Section 2.1.8). In some cases, the agent's goal may not be to maximise their utility, but instead, they aim to achieve a utility that is higher than that of their opponent (see Section 2.1.9).

2.1.2 Negotiation Issues

In a negotiation, the issues are the aspects that the parties need to reach agreement on. For example, in the scenario given in Chapter 1, the issues are the price, battery capacity, condition and colour of the car. Some simple negotiations may involve only one issue (we refer to these as *single-issue* negotiations). If there is more than one issue that is being negotiated over (such as in our scenario), we refer to this as *multi-issue* negotiation.

A multi-issue negotiation takes place over a set of n issues, $i \in I$. We define an offer, o, as a set of values, v_i, one for each issue i in the domain. Formally:

$$o = \langle v_i \rangle_{i \in \{1..n\}} \tag{2.1}$$

where $v_i \in V_i, \forall i \in \{1..n\}$ and n is the number of issues. V_i is the set of values that issue i can take.

Each issue can be classified as being either *continuous* or *discrete*, with discrete issues being further classified by whether or not they have a common ordering.

- **Continuous Issue:** A continuous issue is one which can take any value in a particular range. For example, the battery capacity may be defined to be a real value which ranges between 5 and 20 kWh.
- **Discrete Issue:** A discrete issue can take a value which belongs to a finite set. They may or may not have a known, common ordering.
  - **With a known, common ordering:** For issues that have a common ordering, both negotiating parties are aware that for two nearby values, the evaluation will be similar. As an example, quality may be defined as taking values that belong to the set {*excellent*, *very_good*, *good*, *average*, *poor*}. A quality of *excellent* is similar to a quality of *very_good*. Therefore, for each agent, the evaluation of *excellent* will be closer to *very_good* than any other label. Issues which take an integer value can also be considered in the same way. Even for an issue where the ordering is known, the direction of any opponents' preferences will be unknown. For example, it would not be known whether an opponent's evaluation of *very_good* would be greater or smaller than for *good*.
  - **Without a common ordering:** In contrast, for an issue such as 'colour', which may be defined as taking values that belong to the set {*black*, *white*, *silver*, *red*, *orange*, *yellow*, *green*, *blue*, *purple*} there may not be a common

ordering. That is, one party may consider red and orange to be similar to each other, therefore having similar evaluations, whilst the other party may consider those two colours to be very different to each other, and therefore have highly different evaluations.

In a negotiation against unknown opponents, the lack of a common ordering for some issues adds significant uncertainty to the negotiation, as it makes it difficult to identify which offers are similar to one another. Such uncertainty is common in many negotiations, such as the one introduced in our scenario in Chapter 1. Therefore, in this work, we focus on this type of issue.

For any set of possible negotiation outcomes, each party will prefer some of those outcomes over others. Their preferences can be defined formally using a utility function.

2.1.3 Utility Functions

A utility function describes an agent's preferences, which allows negotiation outcomes to be evaluated and compared. Utility functions may be cardinal or ordinal:

Cardinal utility: Cardinal utility functions map each possible outcome to a real number. For example, outcome A could be valued at 0.5, outcome B at 0.6 and outcome C at 0.7.

Ordinal utility: In contrast, ordinal utility functions provide only an ordering on the negotiation outcomes. For example, an ordinal utility function could simply state that outcome A is preferred over outcome B, and also that outcome C is preferred over outcome B. This differs from cardinal utility in two ways. First, there is no measure of how much outcome A is preferred over outcome B. Second, the ordering can be incomplete. For example, it does not have to state which outcome is preferred between outcome A and outcome C. Ordinal utility functions are considered to be important in situations where a cardinal utility function is difficult to determine. Moreover, it is often easier for humans to express ordinal functions, as it simply involves an ordering over the values, rather than assigning a cardinal value to each one. However, it may be difficult for them to provide a full ordering over all possible outcomes, and therefore in some cases, an incomplete ordering is produced.

Despite the benefits to using ordinal utility functions, we take an approach that is common in the negotiation literature, which is to use cardinal utility functions. The advantage to this is that cardinal utility functions make it possible to measure not only whether or not outcome A is preferable to outcome B, but also by how much. In the multi-issue case, this makes it much easier to identify a set of outcomes that have an equal utility, allowing trade-offs to be made.

The formalisation of a cardinal utility function depends on whether or not the negotiation issues are considered to be independent of each other, as we will now discuss.

Interdependent issues: In some negotiations, there may be issues that are interdependent. Examples include issues that are complements or substitutes of each other. If two issues take values (v_1 and v_2) which are complementary then:

$$U_p(\langle v_1, v_2 \rangle) \geq w_{p,1} \cdot U_p(\langle v_1, \varnothing \rangle) + w_{p,2} \cdot U_p(\langle \varnothing, v_2 \rangle) \tag{2.2}$$

where $w_{p,i}$ is the weight of issue i to agent p and U_p is the utility function of agent p.

For example, in a car sales scenario, complementary issues include the quality of the car stereo and the quality of the speakers. To a buyer who enjoys listening to music, the utility for a car with both a good stereo and good speakers will be higher than the sum of the utilities of the good stereo and the good speakers. On their own, the good stereo or good speakers do not give much utility.

Alternatively, if two issues take values (v_1 and v_2) which are substitutes then:

$$U_p(\langle v_1, v_2 \rangle) \leq w_{p,1} \cdot U_p(\langle v_1, \varnothing \rangle) + w_{p,2} \cdot U_p(\langle \varnothing, v_2 \rangle) \tag{2.3}$$

where $w_{p,i}$ is the weight of issue i to agent p and U_p is the utility function of agent p.

Using again the car sales scenario as an example, substitutes would be whether or not the car has air conditioning and whether or not the car is a convertible. To a buyer who likes to keep cool in the summer, the utility provided by those two issues will be lower for a car which has both of them than the sum of the utilities of the air conditioning and the convertible. Having both of these features gives little more utility than having just one of them.

Independent issues: In contrast, if the issues are independent, the overall utility of the offer is equal to the weighted sum of the utilities of the issues. We refer to such utility functions as being additive, and they can be defined formally as:

$$U_p(o) = \sum_{i=1}^{n} w_{p,i} \cdot U_{p,i}(v_i) \tag{2.4}$$

where n is the number of negotiation issues, $w_{p,i}$ is the weight of issue i to agent p and $U_{p,i}(v_i)$ is the utility to agent p of issue i for the value of v_i. Utility functions that contain interdependent issues cannot be expressed in this form.

The utility functions are typically normalised such that all possible outcomes have a utility in the range $[0, 1]$ and furthermore, for each party, there is an outcome with a utility of 1. Formally:

$$\forall o \in O, \forall p \in P, U_p(o) \in [0, 1] \tag{2.5}$$

$$\exists o \in O, \forall p \in P, U_p(o) = 1 \tag{2.6}$$

In our work, in common with much of the negotiation literature, we consider only utility functions that are additive, and normalised as described above.

So far in this section, we have discussed different ways to determine the utility of an agreement. In any negotiation, there is a further outcome that is possible, which is the lack of agreement. We refer to such lack of agreement as a conflict, which may offer a utility to the participants.

Utility of conflict: In some negotiations, participants may obtain a utility from negotiations that do not result in agreement, which we refer to as the utility of conflict. In the scenario presented in Chapter 1, Bob is able to buy his friend's car, so the utility of conflict would be the utility that he places on this outcome. If the utility of conflict is non-zero, then there may be some agreements that are unfavourable, as the participants can obtain a higher utility by refusing to accept an offer that is worth less than the utility of conflict.

So far in this section, utility has been considered simply as a function of the outcome of the negotiation. However, in some negotiations, the utility can also be affected by time constraints. For example, if two parties negotiate over the sale of a perishable item, its value is likely to decrease over time. Therefore, we now discuss a number of different types of time constraint.

2.1.4 Time Constraints

Negotiation can be affected by time in a number of ways, as discussed by Livne (1979). The effects of time that are commonly considered in the literature are:

Deadlines represent a point in time by when the negotiation must be completed. When the deadline is reached, the negotiation terminates, resulting in disagreement, with each player achieving the utility of conflict.

Discounting represents the impatience of the participants. The more impatient the player, the higher the discounting factor. It has the effect that an agreement that is reached immediately will be preferred over a future agreement that offers the same benefits. Formally, this is modelled using a discounting factor $\delta \leq 1$, where the discounted utility is given by the formula:

$$D(u, t) = u \cdot \delta^t \tag{2.7}$$

where u is the original (undiscounted) utility, and t is the time of agreement. (δ^t is the discounting factor δ raised to the power t.) Note that if $\delta = 1$, there is no discounting and time has no effect on the utility.

Bargaining Costs represent the costs of negotiation itself. A bargaining cost is a fixed cost of making an offer. An example would be a communication cost to each offer that is made. The model that is used is:

$$D(u, t) = u - c \cdot t \tag{2.8}$$

where u is the original (undiscounted) utility, c is the cost of each offer and t is the time of agreement.

In our work, we consider both deadlines and discounting factors (of the form given in Equation 2.7). Without loss of generality, we scale the values of t such that, during the negotiation, $0 \leq t \leq 1$, and therefore in all negotiations, the deadline occurs at time 1.

In most of the existing literature, time is measured as the number of interactions that have occurred (Coehoorn and Jennings, 2004; Fatima et al., 2001, 2006; Lai et al., 2006; Nguyen and Jennings, 2003; Rubinstein, 1982; Yasumura et al., 2009; Yoshikawa et al., 2008). Another way to consider time is to measure the amount of real time that has elapsed, regardless of how many negotiation steps were made. Since automated negotiation allows many offers to be made in a short period of time, the actual number of offers that are made should not have a significant impact on the value of the result. Consequently, the bargaining cost, c, is negligible. What is more significant is the amount of real time that has elapsed, as this has an impact on the parties that the negotiating agents represent. In our scenario, Bob is keen that an agreement is reached in a short time, but it does not matter to him how many offers are exchanged. For this reason, in our work, we choose to consider real-time constraints as these are more realistic and appropriate for real world automated negotiation. Furthermore, such constraints represent an additional challenge for the design of a negotiating strategy, as the number of negotiation steps is no longer fixed, and is dependent on the behaviour of both parties. Additionally, there has been only limited research into this aspect of negotiation. For example, Sandholm and Vulkan (1999) consider negotiation with real-time constraints, but unlike in this work, they only consider single-issue negotiation.

2.1.5 Scenarios and Preference Profiles

Each participant in a negotiation has its own preference profile, Ξ_p, which is formed from the participant's utility function, U_p, their discounting factor, δ_p and their utility of conflict, $U_{\overline{\alpha},p}$. It can be used by the participant to calculate the utility of an offer, at any time during the negotiation. In the negotiations we consider, where any player's utility function can be written as an additive utility function (see Equation 2.4), their preference profile can be defined as:

$$\Xi_p = \langle \{U_{p,i}\}_{i \in \{1..n\}}, \{w_{p,i}\}_{i \in \{1..n\}}, \delta_p, U_{\overline{\alpha},p} \rangle \tag{2.9}$$

where $U_{p,i}$ is player p's utility function for issue i, $w_{p,i}$ is the weight of issue i to player p, δ_p is player p's discounting factor and $U_{\overline{\alpha},p}$ is player p's utility of conflict. We will discuss the forms that $U_{p,i}$ can take in Section 2.2.1.

Furthermore, a scenario, Ξ, consists of a preference profile for each participant in the negotiation, formally defined as:

$$\Xi = \{\Xi_p\}_{p \in P} \tag{2.10}$$

2.1.6 Incomplete Information

In many negotiations, there may be information about the opponent which is unknown. Specifically, the opponent's utility function, negotiation strategy and time constraints may be unknown, as the opponent may not be willing to reveal this information. In contrast, in a negotiation where each agent has complete information about its opponent, it is possible for either agent to determine which outcomes are high performing for both parties (see Section 2.2 for more on performance criteria). Therefore it is easier for the agents to find a good solution. By contrast, in a negotiation with incomplete information, it is impossible for one agent to know exactly how its actions affect the opponent, as this would depend on information that is unknown. In our work, we consider negotiations where the opponent is unknown (Requirement 3) and therefore, the strategy that we develop needs to be able to negotiate without knowledge of the opponent's utility function or strategy.

2.1.7 Negotiation Protocol

The protocol defines the rules of the negotiation, including the types of participants, the negotiation states, the actions that cause the negotiation state to change and the actions that the participants can make in each state (Jennings et al., 2001).

Examples of common negotiation protocols are the *simultaneous offers*, *monotonic concession*, *ultimatum game*, and *alternating offers* protocols. We discuss these in turn.

Simultaneous Offers: Using a simultaneous offers protocol, both parties make their offers at the same time. The *Nash demand game* (Nash, 1953) is an example of a single-shot simultaneous game, where the two parties simultaneously make a single offer. For example, consider a negotiation in which the two parties negotiate over how to share a single pie. Players p and q make a single offer each, $o_p \in [0, 1]$ and $o_q \in [0, 1]$, specifying how much of the pie they wish to take. If their offers are compatible, in this case meaning that the total amount of pie the players specified in their offers did not exceed one ($o_p + o_q \leq 1$) then an agreement is reached. However, it is possible that such an agreement is not efficient, if some of the pie remains unallocated ($o_p + o_q < 1$). Otherwise, if the offers are incompatible ($o_p + o_q > 1$), no agreement is reached, and the negotiation ends in conflict. We discuss efficiency in more detail in Section 2.2.3.

Monotonic Concession Protocol: The monotonic concession protocol, as defined by Rosenschein and Zlotkin (1994), is another example of a simultaneous offers protocol. It differs from the Nash demand game in that if the pair of offers are not compatible, the negotiations continue to another round. Agents are not allowed to make an offer which provides a lower utility to the opponent than the utility of previous offers. That is,

$$U_q(o_{p,r+1}) \geq U_q(o_{p,r}) \tag{2.11}$$

where $U_q(o_{p,r+1})$ is the utility to the opponent (player q) of the offer our agent (player p) made at round $r + 1$ and $U_q(o_{p,r})$ is the utility to the opponent of the offer our agent made at round r.

In addition, if, in a particular round, neither agent concedes, the negotiation ends in conflict. This ensures that the process is guaranteed to terminate, either by repeated concession until an agreement is made, or through a conflict due to lack of concession by both parties. We will look at some of the problems with this approach when we discuss concession strategies in Section 2.3.2.1.

In addition to negotiation protocols that involve the two parties making their offers simultaneously, there are protocols which involve sequential actions, which we will now introduce.

Ultimatum Game: The ultimatum game is a very simple negotiation protocol, in which one party makes an offer and the other chooses to either accept or reject it. In Section 2.3.1.2 we will discuss the limitations of such a simple protocol. A more complex protocol which involves multiple negotiation rounds is the alternating offers protocol.

Alternating Offers: Under the alternating offers protocol, the parties take it in turns to make offers and counter-offers. This continues until one of the parties accepts the opponent's offer or, alternatively, one of the parties chooses to end the negotiation without agreement, or the negotiation deadline is reached. Compared to the ultimatum game, the alternating offers protocol can allow more fair agreements to be reached, since an agent can learn about its opponent's behaviour through the repeated interaction with it. Compared to the simultaneous offers protocol, the alternating offers protocol is more appropriate for many automated negotiation situations, particularly where there is incomplete information. This is since, using a simultaneous protocol, it would be difficult to enforce that the offers of the two parties are compatible and Pareto-efficient without a coordination signal. This coordination signal can be seen as a form of centralised control. Therefore, in our work, we develop an agent which negotiates using the alternating offers protocol.

2.1.8 Negotiation Strategy

A negotiation strategy dictates the approach to negotiation that should be taken by a single agent. Specifically, it specifies the procedure that the agent should use in order to decide what offers to make. This procedure may be based on a number of criteria, including the set of issues being negotiated over, the agent's utility function, the amount of time that has elapsed and the observed behaviour of the opponent. We will discuss specific negotiation strategies in more detail in Section 2.3. The aim of this work is to develop a negotiation strategy that meets the requirements given in Section 1.2.

2.1.9 Spiteful Behaviour

Commonly, the aim of a negotiation is to maximise the utility achieved. However, in some situations, the aim may be to 'win' a negotiation, by achieving a higher utility than its opponent. In such a situation, there is a risk involved in reaching lower utility agreements, as they are more likely to result in other participants achieving a higher utility than our own. As a result, an agent that aims to win a negotiation will need to take a more *spiteful* approach.

In work on spiteful bidding in auctions (Brandt et al., 2007), the spiteful utility, S_p, of agent p is given by:

$$S_p = (1 - \alpha_p) \cdot U_p - \alpha_p \cdot \sum_{q \in P,\ q \neq p} U_q \tag{2.12}$$

where U_p is the utility of agent p, U_q is the utility of opponent q, and $\alpha_p \in [0, 1]$ is the *spitefulness coefficient*, which affects the spitefulness of the strategy. If the spitefulness coefficient, α_p is 0, the agent will be self-interested, with its aim being to maximise the raw utility. At the other extreme, if $\alpha_p = 1$, the agent's goal will be to minimise the opponent's scores, regardless of the effect that has on their own score.

By introducing a spitefulness function, and designing an agent which maximises this adjusted utility, it is possible to change the behaviour of the agent by adjusting the spitefulness function. A more spiteful strategy would result in slower concession, as the agent will regard lower utilities to be of even lower value than their true value.

2.2 Evaluation Criteria and Methodologies

Having introduced the basic notions that are used in this work, we now introduce the methodologies that we use in our evaluation (in Chapters 4 and 6) in order to test our negotiation strategies and the outcomes reached. We first introduce the Genius negotiation environment, which we use as a platform on which to evaluate our strategies (Section 2.2.1), and the Automated Negotiating Agent Competition, which we use as a source of other state-of-the-art negotiating strategies which our strategies can be compared to (Section 2.2.2). We then introduce Pareto optimality (see Section 2.2.3) which we use as a measure of efficiency (Requirement 2). Finally, we introduce the technique of empirical game theory, which we use as a further evaluation method to demonstrate the stability of our strategies in a wide variety of tournaments containing different mixtures of strategies (Section 2.2.4).

2.2.1 Genius Negotiation Environment

In this section, we briefly describe the *Generic Environment for Negotiation with Intelligent multi-purpose Usage Simulation* (Genius) (Hindriks et al., 2009a), which provides a framework for the development of negotiating agents. It facilitates the running of negotiation sessions, under different scenarios and protocols, with a variety of participating agents. Under the alternating offers protocol (as described in Section 2.1.7) provided by the framework, each negotiation session consists of a negotiation between two agents, over a single scenario, which consists of a domain and a corresponding set of preferences.

In this context, a domain specifies the number and types of issues that are negotiated over by the agents. The environment provides support for scenarios containing continuous issues and discrete issues, with and without a known, common ordering (see Section 2.1.2).

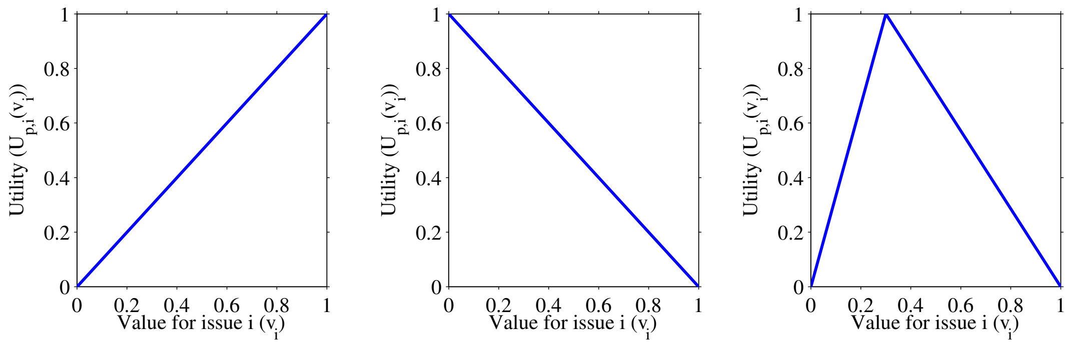
In Section 2.1.5, we introduced the concept of a preference profile, Ξ_p, of the form:

$$\Xi_p = \langle \{U_{p,i}\}_{i \in \{1..n\}}, \{w_{p,i}\}_{i \in \{1..n\}}, \delta_p, U_{\overline{\alpha},p} \rangle \tag{2.13}$$

where $U_{p,i}$ is player p's utility function for issue i, $w_{p,i}$ is the weight of issue i to player p, δ_p is player p's discounting factor and $U_{\overline{\alpha},p}$ is player p's utility of conflict. In the Genius environment, the types of utility functions for each individual issue (denoted $U_{p,i}$) can be classified as one of the following:

- **Linear:** The utility function is linear, and can either be increasing or decreasing as the value of the issue increases. For increasing utility functions, the utility at the lower limit is 0, and is 1 at the upper limit (see Figure 2.1(a)). For decreasing utility functions, the utility at the lower limit is 1, and is 0 at the upper limit (see Figure 2.1(b)).
- **Triangular:** The utility function is triangular, having a single peak. The function is maximised at a particular value of the issue. At this maxima, the utility is 1. On either side of the maxima, the utility decreases linearly to 0 at the upper and lower limits (see Figure 2.1(c) for an example with a peak at 0.3).
- **Discrete:** For a discrete issue, the utility function is a mapping from each possible value of the issue to a utility value. The utility value for each possible value is normalised such that it lies in the range $[0, 1]$ and there is value which gives a utility value of 1.

The negotiation protocol that is used in the Genius environment is the alternating offers protocol (as described in Section 2.1.7), with each offer representing a complete



(a) Uphill (b) Downhill (c) Triangular

FIGURE 2.1: Utility functions used for continuous issues.

package, in that it specifies the values for all issues in the domain. For each session, a deadline is imposed, which consists of a limit of three minutes of negotiation time. In version 3.0 of the GENIUS environment, this was measured independently, per agent, allowing each agent up to three minutes of time. From version 3.1 onwards, this is a single limit, shared by the two agents. Our strategy is designed for use with a single, shared limit, as this type of constraint is more common in negotiation, including in the scenario we introduced in Chapter 1, where the two parties aim to reach agreement by a specific point in time. In either case, if the agents reach their deadline without having formed an agreement, the session ends and the agents receive their utility of conflict. The environment does not impose a limit to the number of negotiation rounds that can take place during the three minute negotiation period.

In addition to the three minute deadline, in order to encourage the agents to reach an agreement in a timely fashion, a discounting factor is applied to the utility generated by an outcome (see Section 2.1.4). The discounted utility, $D(\cdot, \cdot)$, used in the GENIUS environment is given by:

$$D(u_p, t_\alpha) = u_p \cdot \delta_p^{t_\alpha} \tag{2.14}$$

where u_p is the original (undiscounted) utility of the outcome for agent p, t_α is the time at which the outcome was reached and δ_p is the discounting factor for agent p, as specified by the preference profile.

In our work, we use GENIUS as an environment in which to test our agent, since the framework it provides is suitable for the agent we have developed, in that it supports negotiation sessions where the opponent's behaviour and utility function are unknown, and where the sessions have real-time constraints. Furthermore, by using a standard framework, our agents can easily be compared against a range of other state-of-the-art negotiation agents that have been implemented using the same framework. We now introduce the Automated Negotiating Agent Competition, which uses the Genius negotiation environment and has therefore encouraged the development of such agents.

2.2.2 Automated Negotiating Agent Competition

The Automated Negotiating Agent Competition (ANAC)(Baarslag et al., 2010), was initially set up jointly by the Delft University of Technology and Bar-Ilan University to facilitate research into bilateral multi-issue closed negotiation. The competition was held for the first time, as *ANAC 2010* at the 9th International Conference on Autonomous Agents and Multiagent Systems. As a result of the competition, a collection of state-of-the-art negotiating agents, negotiation domains, and preference profiles has been made available to the research community. These form a valuable resource in that they have been developed independently, and represent a varied set of negotiation opponents and settings, making them ideal for benchmarking our strategies against. ANAC 2010 used version 3.0 of the Genius platform as its negotiation environment. In 2011, the competition ran again (as *ANAC 2011*), this time hosted by Nagoya Institute of Technology, using version 3.1 of the Genius platform, which differs slightly from the earlier version, as described in Section 2.2.1. In 2012, we were responsible for running the competition (as *ANAC 2012*), using version 3.2 of the Genius platform, and for the first time, included non-zero utilities of conflict.

For the first competition, ANAC 2010, the organisers developed three negotiation scenarios. For the 2011 and 2012 competitions, further scenarios were developed. After ANAC 2010, it was decided that, for ANAC 2011 and ANAC 2012, the scenarios would be developed by the participants, with each participant entering both an agent, and a scenario into the competition. In ANAC 2011, 18 scenarios were submitted to the qualifying round, with only those belonging to the 8 finalists being used in the final. Of all the competitions, ANAC 2012 used the most extensive set of scenarios. 17 scenarios were submitted by the participants, which were combined with all (non-duplicate) scenarios from ANAC 2010 and ANAC 2011 (final round), to create a total of 24 scenarios. The full set of scenarios were used in both the qualifying and final rounds. The utility functions and outcome spaces of all 24 scenarios are presented in Appendix A.

In all three editions of the competition, for each scenario, a tournament was held, where each agent negotiated against all other agents. The tournament score for an agent was then calculated by taking the average utility that agent achieved in all of the negotiations it took part in, across all scenarios. In each competition, the winning agent was the one which achieved the highest tournament score.

2.2.3 Pareto Optimality

In any negotiation, there are many different outcomes that can be reached. An outcome is considered to be Pareto optimal (or Pareto efficient) if there is no other outcome that would increase the utility of one participant without reducing the utility of another participant (Raiffa, 1982). Such an outcome is efficient as no utility is 'wasted'. Formally, an outcome, o, is Pareto optimal if the following holds:

$$\nexists o^* \in O, (U_p(o^*) \geq U_p(o) \ \&\ U_q(o^*) > U_q(o)) \ \| \ (U_p(o^*) > U_p(o) \ \&\ U_q(o^*) \geq U_q(o)) \tag{2.15}$$

Alternatively, a Pareto optimal outcome can be seen as one for which there is no other outcome that a single participant could select without another participant objecting (Wooldridge, 2009).

In a single issue game, an outcome is considered to be Pareto optimal if it results in all of the resource being allocated. For example, in a game where two participants negotiate over the splitting of a pie of fixed size (Osborne and Rubinstein, 1990), the outcome is Pareto optimal if and only if the whole pie is allocated (that is, if the sum of the shares of the pie is equal to one).

We illustrate the concept of Pareto optimality in bilateral negotiation in Figure 2.2. The axes represent the utilities of each participant. The dots represent the set of all possible outcomes in a given scenario (which consists only of discrete values). The line is the Pareto frontier, which connects all of the Pareto optimal agreements.

As part of our evaluation (in Section 4.5), in order to check that our strategies reach efficient agreements (Requirement 2), we will measure the average distance from each agreement point to the Pareto frontier.

2.2.4 Empirical Game Theoretic Analysis

A strategy which achieves a high tournament score is likely to be good at reaching reasonable agreements with a wide range of other strategies. However, this does not mean that the strategy is the best one to use in all tournaments. It is possible that there is another strategy in the tournament, which achieved a lower tournament score, but which would have performed much better in a tournament where the players used a different mix of strategies.

Therefore, we now consider an analysis technique, known as empirical game theoretic analysis, which can be used to evaluate large games (Reeves, 2005). The idea behind the

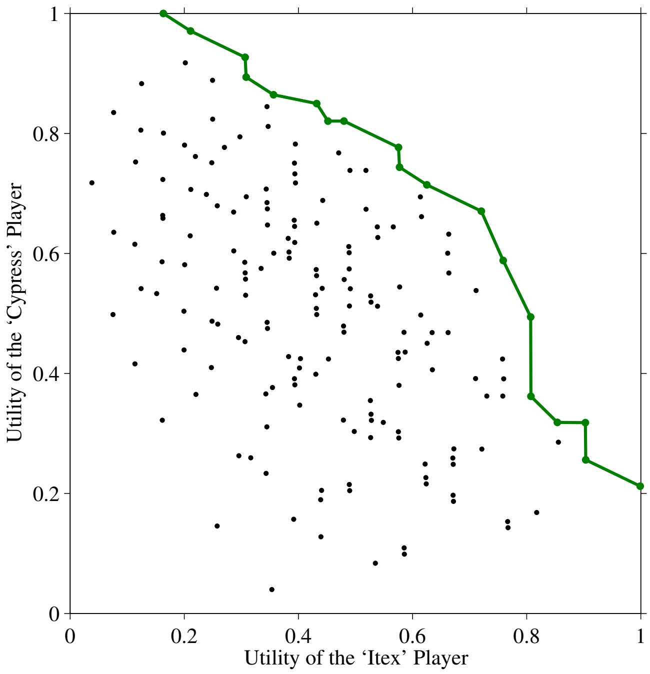


FIGURE 2.2: The outcome space and Pareto frontier from the Itex vs Cypress scenario (detailed in Section A.45). The dots represent the outcome space, and the line represents the Pareto frontier, which connects the Pareto optimal outcomes. The Pareto frontier is not smooth as this is a discrete domain.

technique is to use empirical results to search for equilibria strategy profiles, in which there is no incentive for any player to change its strategy.

Jordan et al. (2007) use this technique to analyse the results of the Trading Agent Competition (TAC). They consider pure-strategy profiles, in which each agent chooses a single strategy, which that agent uses in all negotiations. In a symmetric game with N players and S strategies, there are $\binom{N+S-1}{N}$ such profiles. In the case of the TAC Supply Chain Management (SCM) competition, where $N = 6$ (and in Jordan et al.'s analysis, $S = 6$), the total number of profiles is quite large (462). Therefore, they choose to reduce the game, considering only 3 players. By considering all possible profiles in this reduced game, they present a deviation analysis, which identifies where there is an incentive for one agent to change strategy, and also shows the profiles which are in equilibria. Furthermore, Shi et al. (2012) use a similar technique to search for equilibria in double auction marketplaces. In our work (Section 4.7), we perform an empirical game theoretic analysis of the results of a range of negotiation tournaments.

2.3 Negotiation Strategies

Negotiation strategies can be broadly classified into game theoretic, heuristic and argumentation based approaches (Jennings et al., 2001). In this section we will discuss each of these in detail, considering the appropriateness of various strategies against the requirements we outlined in Section 1.2.

2.3.1 Game Theoretic Approaches

Game theory studies behaviour in strategic situations. It defines a game by its players, actions and payoffs. Specifically, bi-lateral negotiation can be considered to be a game, played by two parties. The actions are the offers that can be made (the exact detail of what an offer consists of depends on the protocol in use). The payoffs are given by each party's utility function.

In addition, game theory defines the extensive-form of a game as a tree, in which the nodes are the points at which decisions can be made (decision points), and the edges indicate the decisions that can be made.

Furthermore, there are two game theoretic approaches to negotiation. These are *cooperative* and *non-cooperative* game theory.

2.3.1.1 Cooperative Game Theory

Cooperative game theory considers games in which it is possible for participants to form coalitions, in order to achieve a greater joint utility than they would if they played the game alone. It considers whether such coalitions are stable, in that there is no incentive for a member of the coalition to leave. It also deals with how the profit from an outcome should be distributed, if the game allows it. In this work, we consider that participants cannot form coalitions, as this adds further complexity to the negotiating environment.

However, even in a negotiation environment where it is not possible to form coalitions, it is possible to use cooperative game theory to characterise the solution space of negotiation problems. We have already discussed Pareto optimality (see Section 2.2.3) which was initially developed in the context of cooperative game theory approach, in order to measure efficiency in such settings. In addition to considering the Pareto optimality of an agreement, cooperative game theory also considers which outcomes are considered to be fair ones. Fairness solution concepts include the *utilitarian* (Myerson, 1981), *Nash* *bargaining* (Nash, 1950a), *Kalai-Smorodinsky* (Kalai and Smorodinsky, 1975) and *egalitarian* (Kalai, 1977) ones. As part of our evaluation, we compare the self-play outcomes with the utilitarian solution as a measure of efficiency (Section 4.4).

2.3.1.2 Non-Cooperative Game Theory

Non-cooperative game theory considers the strategies that can be used by the negotiation participants (Binmore, 1992). It also considers the protocols that are used in negotiation, such as *simultaneous offers*, the *ultimatum game* and the *alternating offers* protocol (see Section 2.1.7). In addition, it defines solution concepts such as the *Nash equilibrium* and the *subgame perfect equilibrium*.

Nash Equilibrium: A outcome is considered to be a Nash equilibrium, if neither participant can benefit by choosing an alternative action, given that all other participants do not change their action (Nash, 1950b).

Subgame Perfect Equilibrium: A subgame is a subset of a game (in extensive-form, see Section 2.3.1) which starts with a single decision node and contains every successor to this node. Another way to consider a subgame is that it is the part of the game that remains at a given point in the game. Now, an outcome is said to be in subgame perfect equilibrium if the outcome is a Nash equilibrium in all subgames of the game. This equilibrium is a refinement of the standard Nash equilibrium, and is a stronger concept (i.e. all subgame perfect equilibria are also Nash equilibria) (Binmore, 1992).

For example, in single-issue negotiation using the ultimatum game (see Section 2.1.7), a subgame perfect equilibrium can be found by reasoning backwards, as follows. Player 2 has a choice of accepting or rejecting player 1's offer. Assuming that the utility gained from conflict is zero, player 2 will accept any offer which results in a utility greater than zero. Therefore, player 1 can make an offer which gives player 2 a utility slightly greater than zero, and player 2 should accept that offer. This game gives player 1 more bargaining power, and therefore the outcome is not fair.

As another example, Rubinstein (1982) studied a game where two players negotiate over the division of a pie of fixed size using the alternating offers protocol (see Section 2.1.7). This is approached with two variations, the first with a fixed bargaining cost, the second with a fixed discounting factor. In both cases, it is assumed that there is no deadline to the negotiation. Rubinstein shows that, with a fixed bargaining cost, the solution which is a subgame perfect equilibrium depends on the bargaining costs of the two players

(Russell and Norvig, 2003). Specifically, if the bargaining cost for player p is c_p, the subgame perfect equilibrium is where player 1 (who makes the first offer) takes:

$$\begin{cases} c_2 & \text{if } c_1 > c_2 \\ x, c_1 \leq x \leq 1 & \text{if } c_1 = c_2 \\ 1 & \text{if } c_1 < c_2 \end{cases} \tag{2.16}$$

Player 2 takes the remainder of the pie.

In an environment where there is a discounting factor (of δ_p for player p) rather than a bargaining cost, the subgame perfect equilibrium is where player 1 takes $(1 - \delta_2)/(1 - \delta_1 \cdot \delta_2)$. Again, player 2 takes the remainder of the pie.

However, this model uses a time constraint which is based on the number of interactions rather than the real-time duration of the negotiation, and it therefore does not meet Requirement 6 (continuous time constraints).

In the work discussed so far in this section, only single-issue negotiation has been considered. When negotiating over multiple issues, there is additional complexity due to the different procedures that can be used.

Procedures for Multi-Issue Negotiation: There are a range of negotiation procedures that define how the single-issue alternating offers protocol can be extended to cover multi-issue negotiation. The procedures that are considered by Fatima et al. (2006) are the *package deal* approach, the *simultaneous* approach and the *sequential* approach. Using the package deal approach, all of the issues are negotiated in a single bundle. With the simultaneous approach, all of the issues are settled simultaneously, but independently of one another. Finally, the sequential approach allows the issues to be negotiated one at a time. It has been shown by Fatima et al. (2006) that, in a less complex setting, in which time constraints are based on the number of interactions, of these three approaches, only the package deal guarantees that equilibrium offers are Pareto optimal. We use the package deal approach in our work, not only due to its Pareto optimality, but also since it is a commonly used approach (Hindriks and Tykhonov, 2008; Lai et al., 2006; Nguyen and Jennings, 2003; Yasumura et al., 2009; Yoshikawa et al., 2008).

Incomplete Information: Harsanyi and Selten (1972) introduce a bargaining model in which there is incomplete information. In their work, uncertainty is modelled by assuming that each participant is of a particular type. The type of the participant represents its utility function, the resources available to it, the amount of information it has, and its beliefs. Each participant knows its own type but not that of its opponent. Rather, its opponent's type is uncertain and given by a probability distribution. Although this approach meets our efficiency requirement (Requirement 2), it does not meet our requirement of being able to negotiate against unknown opponents (Requirement 3), since it requires the a priori probability distribution over the different types to be known.

A further limitation of game theoretic approaches is that they tend to assume full rationality. One of the features of full rationality is that each party has the ability to make whatever calculations are needed to discover the optimal action, however complicated those calculations may be (Rubinstein, 1998). In the negotiation setting that we consider, game theory is a useful tool. However, it is not sufficient in itself because, due to this full rationality assumption, it would require unlimited computational power and therefore would not be considered to be computationally tractable (Requirement 7). We consider game theory to be important in the evaluation of the outcomes of negotiation, and therefore we use Pareto optimality in our evaluation in Chapters 4 and 6. However, in order to find a strategy for which meets all of our requirements (specifically, Requirements 3 (unknown opponents) and 7 (computational tractability)), we consider heuristic approaches.

2.3.2 Heuristic Approaches

In this section we review heuristic approaches to negotiation. Heuristic approaches are based on more realistic assumptions than game theoretic approaches. For example, heuristic approaches do not make the assumption that the opponent is rational. This is an important feature, since our work concerns opponents whose behaviour is unknown (and may therefore be irrational). The aim of heuristic approaches is to produce good, rather than optimal solutions. As such, they are particularly appropriate in complex environments where fully optimal approaches may not be able to provide computable solutions.

We discuss heuristic approaches that can be used to select the desired utility level in Section 2.3.2.1 and then consider methods for learning the opponent's preferences and for predicting future behaviour in Section 2.3.2.2. In Section 2.3.2.3 we discuss methods for making trade-offs between issues without learning the opponent's preferences.

2.3.2.1 Concession Strategies

When there are two agents which aim to divide a resource (consider the split the pie game mentioned in Section 2.2.3), using a protocol that allows for repeated offers to be made (such as the alternating offers protocol described in Section 2.3.1.2), the agents need to choose the rate at which they move away from their preferred offer, towards one which is preferred by their opponent. We refer to this movement as concession. In order for the agents to reach an agreement within the required time frame, it is necessary for them to find an appropriate concession strategy. The concession strategy should ensure that each agent does not concede too quickly, as this results in them giving away too much utility to their opponent. However, if neither agent concedes, they cannot reach an agreement and so this behaviour is also undesirable. We now look at a number of concession strategies that have been proposed in the literature, namely the *Zeuthen concession strategy*, *time dependent strategies* and *tit-for-tat strategies*.

Zeuthen Concession Strategy: When negotiating under the monotonic concession protocol (see Section 2.1.7), the agents need to take care not to stand still (where neither agent concedes at a particular step), as this can result in a conflict even when there exist agreements that are more efficient than conflict. On the other hand, it is unstable for an agent to concede at every step. If it were the case that agent A conceded at every step, and agent B was aware of this, agent B would simply stand still at every step (Rosenschein and Zlotkin, 1994). Consequently, an approach in which both agents concede cannot be a Nash equilibrium, as one of the parties can improve their value of the outcome by changing their strategy.

Rosenschein and Zlotkin (1994) therefore propose the Zeuthen Concession Strategy as an approach which can be used for negotiating using the monotonic concession protocol. It uses the risk evaluation criteria proposed by Zeuthen (1930) to decide which party should concede. In Zeuthen's work, the risk to player p at time t is defined as:

$$K_p^t = \begin{cases} \dfrac{U(o_{p,t}) - U(o_{q,t})}{U(o_{p,t})} & \text{if } U(o_{p,t}) \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.17}$$

where $U(o_{p,t})$ is the utility of the offer $o_{p,t}$ made by player p at time t. $U(o_{q,t})$ is similarly defined for player q.

The agent that should concede is the one which stands to lose the most from conflict, and is therefore the one that is least willing to risk conflict (and has the lowest value of K_p^t). A possible limitation of this approach is that it requires complete information about the opponent's utility function, in order to determine which agent has the lowest value of K_p^t. If an estimate is used, there is a chance that the agents could both consider their value of K_p^t to be the highest, and therefore neither agent would concede, resulting in a conflict.

We therefore consider other approaches which do not require the opponent's utility function to be known.

Time Dependent Concession Strategy: Using time dependent concession (Faratin et al., 1998), the utility level is calculated as a function of time. Commonly, this function is either polynomial, or exponential.

Using a polynomial function, which is a standard choice made in much of the negotiation literature (Faratin et al., 1998; Fatima et al., 2001), the desired utility level $U(t)$ at time t is given by:

$$U(t) = U_0 - (U_{\min} - U_0) \cdot t^{1/\beta} \tag{2.18}$$

where U_0 is the initial utility, $U_{\min}$ is the reservation utility (which the agent will not concede beyond) and β is the parameter that affects the rate of concession. We can partition the β value into three types:

- **Boulware** or **Tough:** ($\beta < 1$) Initially the agent concedes very little, but increases the rate of concession as the game progresses.
- **Linear:** ($\beta = 1$) The agent concedes at a constant rate throughout the duration of the negotiation, reaching its reservation utility at the deadline.
- **Conceder** or **Weak:** ($\beta > 1$) The agent concedes quickly at the start of the negotiation, with the rate of concession slowing as the game progresses.

Figure 2.3 shows the utility levels U over time t for three different values of β, where $U_0 = 0.95$ and $U_{\min} = 0.5$.

Using such a strategy, it is necessary to choose the value of β in order to concede at an appropriate rate. Additionally, we need to find a way to set $U_{\min}$. Alternatively, we can use a tit-for-tat approach, which aims to concede at a similar rate to the opponent.

Tit-for-tat Concession Strategy: Under the tit-for-tat concession strategy, the agent chooses its concession based on that of its opponent in the previous round (or rounds). In this context, Faratin et al. (1998) propose a number of variations of a tit-for-tat based approach.

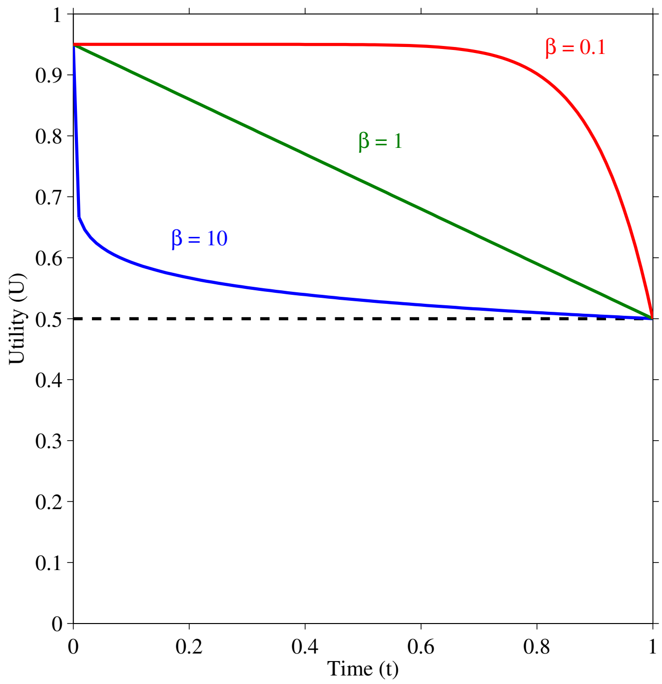


Figure 2.3: Utility levels according to various concession rates ($\beta = \{0.1, 1, 10\}$) with initial utility $U_0 = 0.95$ and reservation utility $U_{\min} = 0.5$ (shown as a dashed line).

Using *relative* tit-for-tat, the agent concedes by the same percentage as its opponent made σ rounds ago. In *random absolute* tit-for-tat, the agent concedes by the same amount as its opponent made σ rounds ago (plus or minus a random amount in the interval $[0, M]$). The random behaviour is introduced to allow the agents to escape from local minima. In *averaged* tit-for-tat, the agent uses the average concession of the opponent's σ previous offers to select its concession amount.

All of these tit-for-tat approaches consider the offers that have been made at a particular round (or set of rounds). Due to our requirement for continuous time constraints (Requirement 6), the number of rounds that have elapsed should not be considered to have an impact on the negotiation, as they do not affect the outcome. Therefore, if we were to use any of these approaches, it would be necessary to adapt them for use with continuous time.

None of the concession strategies that we have discussed take any discounting factor into consideration. In our work, we use a concession strategy that is based on the time dependent strategy, but in common with the tit-for-tat strategy, we consider our opponent's offers. The aim is to reach an efficient agreement (Requirement 2) by optimising our rate of concession as a best response, based on observations of the opponent's offers, and considering the discounting factor. In more detail, if our concession is too slow, it will take longer to reach an agreement, and utility will be wasted due to the effect of discounting. Alternatively, if our concession is too fast, we will easily reach a quick agreement, but with low utility.

2.3.2.2 Learning Techniques

In a negotiation where there is uncertainty about the behaviour or the preferences of the opponents, it is often necessary to use a learning technique in order to estimate how the opponent will behave in future and/or to estimate the preferences of the opponent. There are a number of different techniques that can be used for this learning. Many of the heuristic approaches to negotiation which we discuss in Section 2.3.2 use the techniques which we introduce here. In this section, we briefly introduce the techniques of *Bayesian updating*, *kernel density estimation*, *least squares regression*, *Gaussian process regression* and *reinforcement learning*, giving examples of their use in automated negotiation.

Bayesian Updating Bayesian updating is the process of using Bayes' theorem in order to update the likelihood of a set of beliefs. Bayes' rule is defined as follows:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.19}$$

where $P(A|B)$ is the posterior probability of A given B, $P(B|A)$ is the likelihood (or the conditional probability of B given A), $P(A)$ is the prior probability of A and $P(B)$ is the prior probability of B.

In the context of automated negotiation, Bayesian updating has been proposed as a suitable method to classify the type of behaviour of an agent, or its utility function (Zeng and Sycara, 1998; Lin et al., 2006; Hindriks et al., 2009b). For example, Lin et al. (2006) have developed an agent which uses Bayesian updating in order to learn the type of the opponent during a single negotiation. To use Bayesian updating, the agent must first choose the prior probability of each type. Specifically, they consider the prior probability $P(\tau)$ of each type τ to be equal, that is $\forall \tau \in T, P(\tau) = 1/|T|$ where T is the set of possible types, and $|T|$ is the size of that set. By Bayesian updating, the posterior probability $P(\tau|o_t)$ can be found as follows:

$$P(\tau|o_t) = \frac{P(o_t|\tau) \cdot P(\tau)}{P(o_t)} \tag{2.20}$$

where $P(o_t|\tau)$ is the probability of offer o_t given that the opponent is of type τ, $P(\tau)$ is the prior probability of type τ and $P(o_t)$ is the probability of offer o_t.

A limitation of this approach is it requires the agent to update the probabilities of a large number of hypotheses (one for each possible type, and the number of possible types can be very large). As a result, it does not scale well in negotiation domains with a large number of issues. In a negotiation consisting of n issues, each issue has one of m weights and one of l possible evaluation functions, the number of types $|T|$ would be $(l \cdot m)^n$.

Bayesian Updating, Scalable in the Number of Issues In an attempt to produce an approach which scales better in the number of issues, Hindriks and Tykhonov (2008) develop an agent which is based on the work of Lin et al.. Rather than learning the weights of each issue, they learn an ordering over the weights (as this reduces the number of hypotheses compared to trying to learn the weights themselves), which they refer to as the *issue priorities*. In addition, their approach treats the utility function for each issue ($U_i(v_i)$) and the issue priorities (w_i) as being independent. Therefore the number of hypotheses that need their probabilities updating is significantly reduced, making their solution more computable in larger domains.

In order to estimate the weight of each issue, based on the issue priorities that have been learnt, their agent considers all possible orderings of the issues as a set of weight hypotheses H^w. From each ranking, the weight is calculated as follows:

$$w_i^h = 2 \cdot \frac{r_i^h}{n \cdot (n+1)} \tag{2.21}$$

where r_i^h is the rank of issue i in hypothesis $h \in H^w$ and n is the number of issues. The set of weight hypotheses for a single issue i can then be denoted by $h_{i,z}^w \in H_i^w$.

In terms of the utility functions for each issue, it is assumed that these can be modelled by taking a weighted average over a set of functions, with each function being *linear increasing*, *linear decreasing* or *triangular*, as defined in Section 2.2.1. Consequently, each of these functions are considered as a hypothesis, and the agent therefore considers a set of hypotheses $h_{i,z}^e \in H_i^e$ for each issue i.

Their agent initially assumes that the probability distributions over each set of hypotheses H_i^w and H_i^e are uniform (unless it has some additional knowledge). As part of the strategies that we have developed, where necessary, we use this scalable approach to Bayesian updating in order to learn our opponent's utility function.

Kernel Density Estimation Kernel density estimation (KDE) is the process of estimating the distribution of a value, using an estimator of the form:

$$P(x) = \frac{1}{N} \sum_{i=1}^{N} K(x, x_i) \tag{2.22}$$

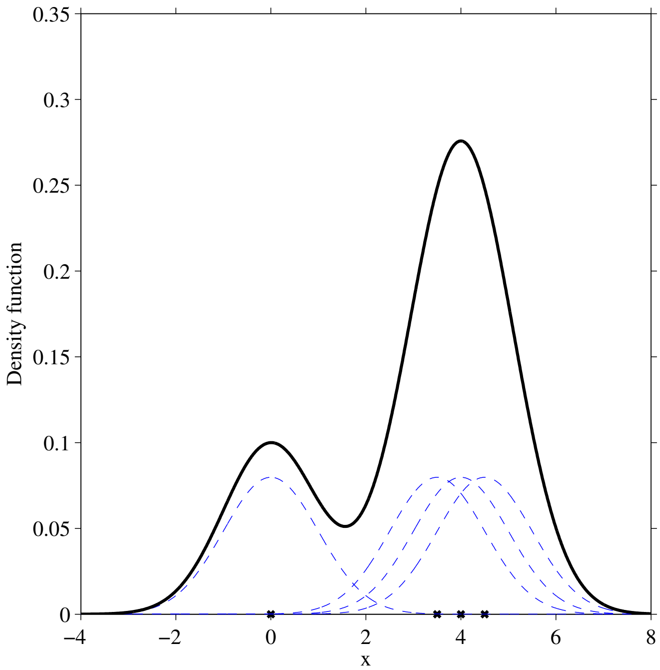
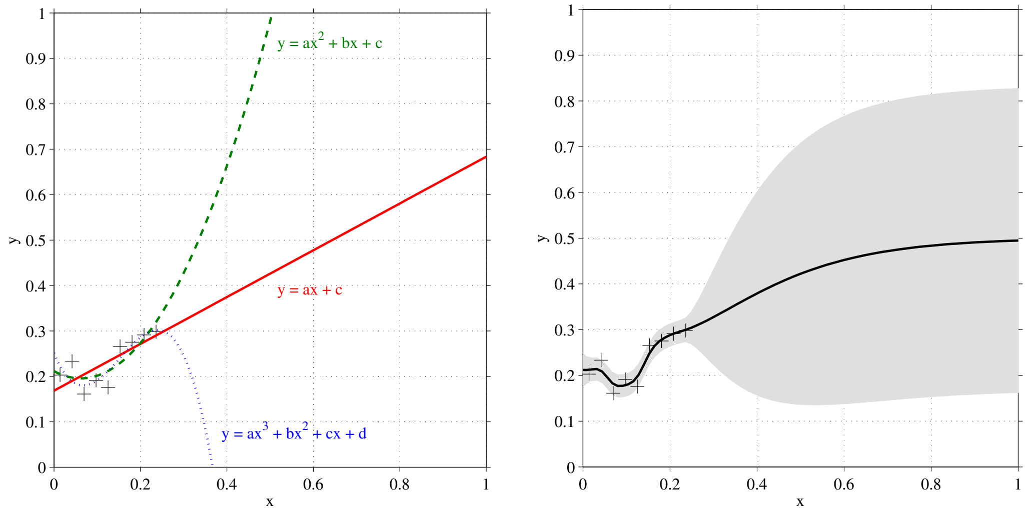


FIGURE 2.4: Kernel Density Estimation for four data points. The data points are marked as crosses, the kernels as dashed lines and the overall estimation as a solid line.

where $K(\cdot, \cdot)$ is the kernel function, which has an integral of 1. Commonly this kernel function is a Gaussian distribution. The aim of kernel density estimation is to build a probability distribution over a set of values. This requires the kernel function's parameter to be chosen appropriately. For a Gaussian distribution kernel function, this is done by adjusting the variance of the distribution. Figure 2.4 shows an example of kernel density estimation performed on a set of data points, for a single variance.

Coehoorn and Jennings (2004) use the KDE approach in order to learn the opponent's preferences. This uses the negotiation history of the agent in a particular scenario, in an attempt to learn the weights associated with each issue. During the negotiation, the process for estimating these weights is a Fourier transform which can be performed in $O(n \log n)$ time (with n being the sampling rate). However, their approach requires the agent to use information from previous interactions, which we consider not to be available as part of Requirement 3 (unknown opponents).

Least Squares Regression Least squares regression is the process of finding a curve which best fits through a given set of points. The curve has a particular function which takes a number of parameters. The quality of the fit is measured by the sum of the squares of the offsets between the points and the solution curve. The solution curve is therefore the one which minimises the squares of the offsets (Weisstein, 1999).



(a) Using least squares regression (using first, second and third degree polynomials)

(b) Using Gaussian process regression (showing the mean and 95% confidence interval)

FIGURE 2.5: Example regression techniques.

Figure 2.5(a) illustrates least squares regression over a set of data points, using first, second and third degree polynomials. These polynomial functions are given by: $y = ax + b$, $y = ax^2 + bx + c$ and $y = ax^3 + bx^2 + cx + d$ respectively. As can be seen in the figure, the prediction varies greatly according to the degree of the polynomial, particularly for values of x that are far outside the range of x in the input data.

Gaussian Process Regression Gaussian process regression is more advanced than least squares regression. The process is defined by the mean function and the covariance function (Rasmussen and Williams, 2006). The mean function describes the expected output, when no relevant input information is available, whilst the covariance function describes how the output varies compared to nearby points.

For example, Figure 2.5(b) shows the Gaussian process regression over a set of data points. The mean function in use is:

$$y = 0.5 \tag{2.23}$$

whilst the covariance function is defined by a matrix.

A benefit of using a Gaussian process regression compared to a linear one is that the output of the Gaussian process is both a mean prediction and measure of the confidence in that prediction. Figure 2.5(b) shows how the confidence in the prediction is greater at points close to the input data. The confidence information is particularly useful for a spiteful strategy, which aims to make high utility agreements in order to get a higher score than its opponent, as it affects the expected spiteful utility. Specifically, an agent using a spiteful strategy will prefer to wait for a later agreement with a similar mean but a larger variance, as the larger variance indicates that the agent may achieve a very high spiteful utility. We make use of Gaussian process regression in one of our strategies, and we use the confidence information in its calculation of a best response to the learnt concession of the opponent. This is discussed in more detail in Section 3.2.2.

When using Gaussian process regression techniques, it is necessary to choose an appropriate covariance function. Examples include the *Matérn* and *squared exponential* covariance functions (Rasmussen and Williams, 2006). Both of these functions are stationary, in that they are based only on the distance between two points. Specifically, the Matérn covariance function is:

$$C(d) = \frac{2^{1-v}}{\Gamma(v)} \left(\frac{2\sqrt{v}d}{l}\right)^{v} K_v \left(\frac{2\sqrt{v}d}{l}\right) \tag{2.24}$$

where d is the distance between the points, $\Gamma(\cdot)$ is the Gamma function (an extension of the factorial function), v and l are parameters of the covariance function and K_v is a modified Bessel function (Rasmussen and Williams, 2006). Furthermore, the squared exponential function is:

$$C(d) = e^{-d^2/v} \tag{2.25}$$

where d is the distance between the points and v is a parameter of the covariance function (Rasmussen and Williams, 2006). The squared exponential covariance function is a special case of the Matérn covariance function and therefore we use the more flexible Matérn covariance function in our work.

Reinforcement Learning Using reinforcement learning, the agent learns based on the rewards it receives from performing a task. Yoshikawa et al. (2008) and Yasumura et al. (2009) use a reinforcement learning approach in order to choose an appropriate concession strategy. However, in order to perform this effectively, the agents are required to conclude many prior negotiations with an opponent in order to learn the opponent's behaviour. Consequently their approach, and reinforcement learning in general, is not appropriate for one-off negotiation with an unknown opponent (Requirement 3).

2.3.2.3 Making Trade-Offs

The approaches discussed above involve opponent modelling, however, it is not always necessary to know the utility function of the opponent in order to make trade-offs. Instead, the similarity between a pair of offers can be measured and used to determine the trade-offs that should be made. By this, we mean that the closeness between the values of the offers are considered, regardless of the utility functions of any party. To this end, there are a few different approaches that can be used by an agent to make trade-offs between different issues without learning the preferences of the opponent.

Faratin et al. (2002) were the first to propose an algorithm for negotiating over multiple issues which makes use of *similarity-based reasoning*, that is, it tries to find offers that are similar to the opponent's previous offer. They use a fuzzy similarity approach to consider the closeness of offers. Their algorithm for trade-offs works as follows: The process starts at the offer, $o_{q,r-1}$, that was previously made by the opponent and by altering the values of that offer, generates N new 'child' offers which each have a utility E greater than the utility $U(o_{q,r-1})$ of the opponent's previous offer. Of these child offers, the one which is most similar to $o_{q,r-1}$ is selected to be the new parent. This process is then repeated a number of times, until the utility of the selected offer matches our desired utility. The number of times is chosen by setting E to be some fraction $(1/z, z \in \mathbb{Z})$ of the difference between $U(o_{q,r-1})$ and our desired utility. Their approach is designed for discrete issues.

The limitation of their approach is that it requires the criteria functions and the weights of each criteria function for any issues that are used in the negotiation scenario to be known. Due to Requirement 5 (unknown ordering), we require an approach which can be used for issues which do not have a known, common ordering, and therefore where such knowledge is unavailable.

To address this, Somefun et al. (2006) take a similar approach to the work of Faratin et al., although they consider trade-offs between continuous issues. Specifically, they present an algorithm for Pareto-search in an environment where there is no prior knowledge of the opponent's preferences, and they show that the algorithm reaches an agreement that is approximately Pareto efficient. Their algorithm works as follows:

1. The agent first chooses the desired utility level, by referring to its concession strategy.
2. The agent then builds a surface which represents all of the points with the desired utility. We refer to this as an iso-utility surface.
3. The agent finds the point on that surface that is closest to the previous offer of the opponent in terms of the Euclidean distance.
4. The agent makes the counter-offer that is represented by that point.

Figure 2.6 shows this process graphically in a scenario with two continuous issues which are represented by the two axes. Here, the two curved lines are iso-utility curves of two agents, at a particular utility level, $U(\{v_1, v_2\}) = 0.82$. Agent A has a preference for low values of both issues (shown by the curve in the bottom left), and agent B has a preference for high values of both issues (shown by the curve in the top right). Furthermore, agent A considers issue 2 to be more important, whilst agent B considers issue 1 to be more important. Specifically, agent A's utility function is:

$$U(\{v_1, v_2\}) = \frac{1}{3}v_1^2 + \frac{2}{3}v_2^2 \tag{2.26}$$

whilst agent B's utility function is:

$$U(\{v_1, v_2\}) = \frac{2}{3}(1 - v_1)^2 + \frac{1}{3}(1 - v_2)^2 \tag{2.27}$$

Figure 2.6 shows the offers that are exchanged by two agents that both use the orthogonal search method. Agent A proposes offer 1 ($\{2, 4\}$), and agent B observes this offer. Agent B finds the point on its iso-utility curve which is closest to that offer, and proposes the counter-offer that is represented by that point, shown as offer 2 ($\{6.6, 6.41\}$). This process is then repeated by agent A. As the agents concede, they will perform this process with different iso-utility curves (since their desired utility level will change).

When both agents use this strategy, once they have both conceded enough that the iso-utility curves of the two agents intersect each other, the offer will be made at the intersection, and the other agent should accept the offer.

We base one of our strategies on this technique, since it meets our requirements for use against unknown opponents (Requirement 3) and it is a computationally tractable approach (Requirement 7). Its limitation is that it does not meet Requirement 5, since it requires the discrete issues to have a known, common ordering. Therefore we extend it to cover such issues.

So far, this section has considered both game theoretic and heuristic approaches to negotiation. There remains a further approach to negotiation, known as argumentation, which we briefly discuss below, for completeness, although it is not directly relevant to the setting considered in this thesis.

2.3.3 Argumentation

In negotiation by argumentation, participants are able to communicate additional information to the other participants. In addition to the offers and counter offers that are used in the alternating offers protocol, critiques and explanations can also be sent. Critiques are comments on which parts of the proposal the agent likes or dislikes. Explanations

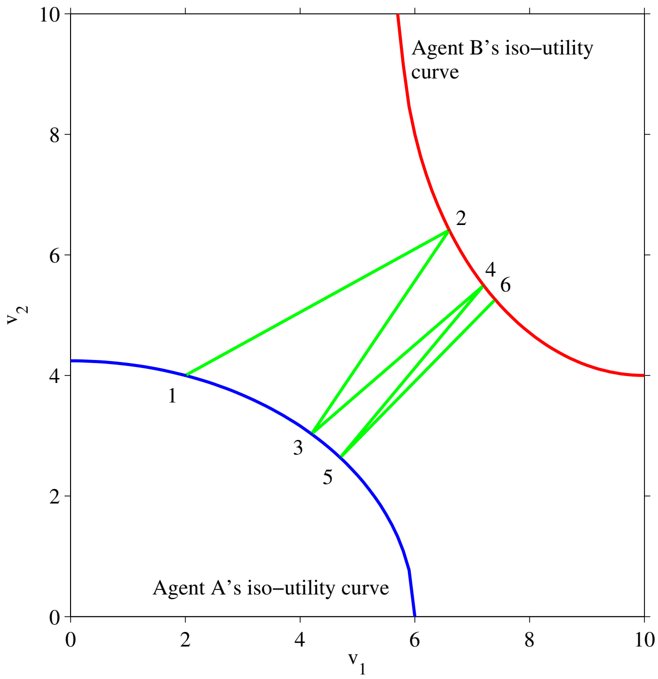


FIGURE 2.6: Demonstration of the Pareto-search algorithm, showing the iso-utility curves of two agents. Agent A proposes offer 1 at $\{2, 4\}$. Agent B finds the point on his curve that is closest to offer 1, and proposes the counter-offer represented by that point, shown as offer 2 at $\{6.6, 6.41\}$. The process is then repeated by each agent in turn.

are a way that agents can support their proposals, and may take the form of threats, rewards or appeals (Parsons and Jennings, 1996; Jennings et al., 2001). The arguments that are exchanged take the form of propositional logic statements, and each agent uses logical proof to evaluate the set of arguments that it has received (Wooldridge, 2009). Argumentation based negotiation has the benefit that it can achieve agreements that other approaches could not, by using arguments to change the preferences of the opponent. However this is at the expense of significant overheads, due to the reasoning that the agent needs to perform in order to evaluate the arguments. It is possible for agents to make arguments that are not truthful, which further complicates the negotiation, since the agent needs to evaluate each argument's credibility (Jennings et al., 2001). Since it would be difficult to evaluate the performance of an agent in an environment where the preferences can be changed, we choose to focus on negotiation where each party's preferences are fixed. Therefore we do not consider negotiation by argumentation in the remainder of this thesis.

2.4 Negotiation with Many Parties

All of the literature that has been reviewed in this section has been focused on negotiation between two parties, and therefore does not meet our requirement for coordination of negotiation with a number of participants (Requirement 8).

To this end, in this section, we review negotiation involving more than two participants. In some environments, the number of participants may not be fixed. For example, in the scenario we gave in Chapter 1, buyers can enter and leave the negotiation at any time. We begin this section by discussing the issues surrounding concurrent bi-lateral negotiation in Section 2.4.1. Then, in Section 2.4.2 we discuss an additional feature of some concurrent negotiation environments, including the one we consider (Requirement 9), known as *decommitment*.

There are two ways in which negotiation with many parties can be carried out. In *sequential negotiation*, each agent negotiates with one opponent at a time, taking it in turns to negotiate with each opponent. In *concurrent bi-lateral negotiation*, the agents negotiate concurrently with a number of opponents. The offers that are made are still bi-lateral in that each offer is made by one party to another. Therefore, all offers and agreements are made between exactly two negotiation partners. In time constrained scenarios such as the one in our scenario, concurrent bi-lateral negotiation is considered to be more appropriate than sequential negotiation, as sequential negotiation can lead to 'lengthy negotiation encounters' (Nguyen and Jennings, 2003). Therefore the next section focuses on concurrent bi-lateral negotiation.

2.4.1 Concurrent Bi-lateral Negotiation

One of the earlier approaches to task allocation amongst a large number of parties is the contract net, proposed by Smith (1980). Here, the process begins with a task announcement which specifies the requirements that a requester needs any bidders to satisfy, along with a brief description of the task, and a specification of the information that is required in a bid. In addition, the requester specifies a deadline by which bids must be received. Upon receipt of a task announcement, each bidder evaluates the requirements and the specification of the task, in order to decide whether or not to bid on the task. When a bidder decides that it should work on a particular task, it sends a bid for that task. The requester evaluates the bids that have been received, and if it has one which it considers to be satisfactory, the bid is accepted and the task is awarded to the bidder. A limitation of this approach is that when there are multiple requesters, the bidders need to decide the order in which to bid on tasks, since they may not be able to complete some combinations of tasks, and therefore they may need wait to see if a bid is accepted before making another bid.

In an extension to Smith's work, Aknine et al. (2004) propose two phases of proposal and allocation to allow an agent to concurrently manage several negotiation processes. Specifically, the process is as follows: the requester makes an announcement (as before), the bidders then respond with *pre-bid* messages. The requester can then send a *pre-accept* message in response to one of the pre-bid messages in order to temporarily accept the pre-bid. The requester sends *pre-reject* messages to all of the other bidders. At this point, the bidder who received the pre-accept message is able to send a *definitive-bid* which the requester is able to *definitive-accept* or *definitive-reject*. The bidders that were sent pre-reject messages are able to send further pre-bid messages. Until the requester has sent a definitive-accept or definitive-reject message, it is able to send further pre-accept and pre-reject messages if it receives more favourable pre-bids. The benefit of this approach over the earlier work of Smith is that it allows for more efficient negotiation in a many to many case, for example, where there are multiple buyers and multiple sellers.

The alternating offers protocol used for two participants has also been extended to allow participants to negotiate with more than one opponent at a time. To this end, Dang and Huhns (2005) propose an alternating offers protocol which is based on the work of Aknine et al. and has two-phase commitment and rejection. It differs from the contract net approach by Aknine et al. (2004) in that the process begins with the alternating offers protocol in which agents exchange offers and counter-offers with each other, rather than a single task announcement being sent.

In another line of work, Nguyen and Jennings (2003) design an agent for concurrent bi-lateral negotiation. In order to concurrently manage a number of negotiations, their approach uses a coordinator, which manages a number of individual negotiation threads. Each negotiation thread handles bi-lateral negotiation with a single opponent. The coordinator is responsible for coordinating the negotiation threads and choosing their strategies. Specifically, in terms of choosing a strategy, the agent attempts to learn whether the opponent is a *conceder* or a *non-conceder*. If the opponent is believed to be a conceder, the agent uses a tough strategy with probability P_t^c. On the other hand, if the opponent is believed to be a non-conceder, the agent uses a conciliatory strategy with probability P_c^n. In addition, the classification of opponents into conceder or non-conceder classes depends on concession by the opponent at every negotiation round. In an environment where there are multiple issues and where our agent's utility function is unknown to the opponent, such perfect behaviour is unlikely to occur, as it relies on the opponent being able to identify the Pareto-efficient offers. Therefore, their approach would need to be considerably extended in order to meet our requirements for a strategy which works against unknown opponents (Requirement 3) and multiple issues. We use their approach, with some fixed parameters, as a benchmark in the evaluation of our many-to-many negotiation strategy in Chapter 6.

2.4.2 Decommitment

One of the additional features of concurrent negotiation environments, including the ones considered in this work (Requirement 9), is the concept of decommitment. In a concurrent negotiation, one of the parties may make an agreement with an opponent, before finding an opportunity for a better agreement with another opponent. Decommitment allows one of the parties to cancel an agreement that it has made, allowing it to select the alternative agreement and therefore achieve it a greater utility. By allowing decommitment, the strategies can be more flexible, allowing agreements with a higher utility to be reached. However, due to the costs of preparing to perform a contract, any unnecessary decommitment (where a party decommits in order to commit to another contract which is only slightly better) could lead to a 'decrease in the sum of utilities of the parties' (Ponka, 2009). It is therefore important that there is some form of penalty for decommitment, in order to discourage unnecessary decommitment. In this section, we consider two approaches to decommitment, namely *contingency contracts* and *leveled commitment contracts*.

Contingency Contracts: Contingency contracts are contracts in which the existence of the contract is tied to future events (Raiffa, 1982). A limitation of this approach is that it requires all possible future events that can affect the contract need to be considered. This may be an unrealistic assumption, as in automated negotiation it would require the participants to express all such future events to the agent. In addition, if there are a large number of possible events, it may not be possible to monitor all of them (Sandholm and Lesser, 2001).

Leveled Commitment Contracts: The leveled commitment contract approach (Sandholm and Lesser, 2001) allows a party to decommit through the payment of a decommitment fee. According to work by Andersson and Sandholm (2001), decommitment fees used in this approach can be:

1. A **fixed value** which is decided prior to the negotiation.
2. A **percentage of the contract price**, with the percentage being decided prior to the negotiation.

3. A **value decided at the time of contracting**, as a percentage of the contract price, with the percentage increasing as the time of contracting increases.

4. A **value decided at the time of decommitment**, as a percentage of the contract price, with the percentage increasing as the time of the decommitment increases.

Using the leveled commitment contract approach, Nguyen and Jennings (2005) develop a strategy for use in an environment which allows decommitment through the payment of a penalty. In their model, the decommitment fee $\rho(t)$ at time t is calculated using the fourth of the methods given by Andersson and Sandholm (2001), specifically:

$$\rho(t) = U(\alpha, t_\alpha) \cdot (\rho_0 + \frac{t - t_\alpha}{1 - t_\alpha} \cdot (\rho_{\max} - \rho_0)) \tag{2.28}$$

where $U(\alpha, t_\alpha)$ is the utility of the agreement at time the contract was made t_α, ρ_0 is the penalty at contract time, $\rho_{\max}$ is the penalty at the deadline. Their strategy is designed for one-to-many negotiations with uncertainty about the opponents. However, it only considers discrete time, makes strong assumptions about the opponents, and requires considerable prior knowledge about these opponents. In particular, they assume that there is a small number of different opponent *types*, all using a simple time-based concession strategy. Furthermore, they assume that the probabilities of each type are known, as well as the payoff that will be obtained when negotiating against each type. In contrast, we consider an environment in which the agents do not have such knowledge.

We will use this form of decommitment fee in our concurrent negotiation environment as, in many scenarios, it is the one which most closely represents the costs involved in decommitment. Specifically, as time progresses from the point of agreement, the costs incurred in fulfilling a contract are likely to increase. Furthermore, the opportunity to find an alternative contract before a deadline (and the utility from it) decreases (Andersson and Sandholm, 2001).

Finally, An et al. (2008; 2011) and An (2011) have developed an agent which negotiates in an environment which contains concurrent negotiation with decommitment. In common with much of the other negotiation literature, their work considers time constraints to be based on the number of negotiation rounds rather than the amount of elapsed time. However, the environment they consider is multi-resource rather than multi-issue. In such an environment, each resource can be negotiated independently, creating a complete package by reaching agreement on each resource with a different opponent. Furthermore, each opponent may offer only a subset of the resources. Their strategy uses time-dependent concession to concede at a different rate for each resource, depending on the relative scarcity of that resource. A related approach is taken by Shi and Sim (2008), who have also developed a further strategy for concurrent multi-resource negotiation with decommitment. In contrast, in our work, the individual issues cannot be split amongst negotiation partners. For example, when a buyer negotiates over the sale of a car, it is not possible to reach agreement by negotiating a good price with one seller whilst agreeing the colour with another.

For these reasons, the strategies developed by Shi and Sim and An et al. are not suitable for the negotiation environment we consider due to the way in which each issue (or resource) can be negotiated separately. Therefore we will use the strategy developed by Nguyen and Jennings as a benchmark which we can compare our strategies against.

2.5 Summary

In this chapter we began by introducing the key notions within the automated negotiation literature. In addition, we introduced a number of evaluation criteria and methodologies that can be used to measure the performance of a negotiation strategy. In doing so, we introduced the efficiency concepts which form Requirement 2. Specifically, in our evaluation in Chapters 4 and 6, we will use Genius as our test environment, and will compare our strategies against those produced for the Automated Negotiating Agent Competition. Furthermore, we will use empirical game theoretic techniques in our evaluation, in order to evaluate the performance of our strategy in tournaments where more than one opponent uses a single strategy.

Subsequently, we reviewed the literature relating to negotiation strategies, considering game theoretic, heuristic and argumentation based approaches. Furthermore, we highlighted the problems involved in using purely game theoretic approaches to meet our requirements, and therefore demonstrated the need to use heuristic strategies for our purpose, since they tend to be more computationally tractable (Requirement 7) even in complex negotiations where the behaviour and preferences of the opponents are unknown (Requirement 3) and multiple issues are present (Requirement 4). Finally, in order to address our requirement for a solution that can coordinate multiple concurrent negotiations (Requirement 8), we discussed existing work relating to negotiation situations where there are more than two participants.

In Chapters 3 and 5, we will develop our own negotiation strategies, which build upon some of the existing work that has been discussed in this chapter. Specifically, our negotiation strategies use time dependent concession to set their desired utility at a given time. As part of this concession strategy, they adapt their rate of concession as a best response to the expected future behaviour of the opponent. Furthermore, in order to predict this future behaviour, they use either a simple least squares regression over the observations of the opponent's offers or a more advanced Gaussian process regression technique which provides a measure of the confidence of the prediction, which can be used to improve the choice of concession rate.

Chapter 3

Design of One-to-One Negotiation Agents

In this chapter, we present the negotiation agents that we have developed for one-to-one negotiation, and the strategies that are used by our agents. The purpose of developing a range of strategies, rather than a single one, is to enable us to consider the benefits of different approaches. Our overall aim is to use a suitable combination of these approaches in order to develop a negotiation strategy which meets all of the requirements outlined in Section 1.2.

In the remainder of this chapter, we provide an overview of our one-to-one negotiation strategies (Section 3.1), before describing in detail each part of the strategies, in turn (Sections 3.2 to 3.6). We then explain how our agents are formed from these strategies, including implementation details (Section 3.7). We summarise the chapter in Section 3.8.

3.1 Overview

Our strategies are designed to participate in multi-issue, bi-lateral negotiation, in which two parties negotiate over multiple issues in order to reach an agreement. The negotiation protocol that is used is based on the alternating offers protocol, which we introduced in Section 2.1.8. In more detail, each offer, o, represents a complete package, in that it specifies a value for all issues. Formally, $o = \langle v_1, v_2, \ldots, v_n \rangle$, where v_i is the value for issue i and n is the number of issues. The possible actions under this protocol are OFFER, ACCEPT and END. The negotiation begins with the agents exchanging OFFER messages. Sending an OFFER message in response to an OFFER from the opponent constitutes a *counter*offer and an implicit rejection of the previous offer. If an agent is satisfied with

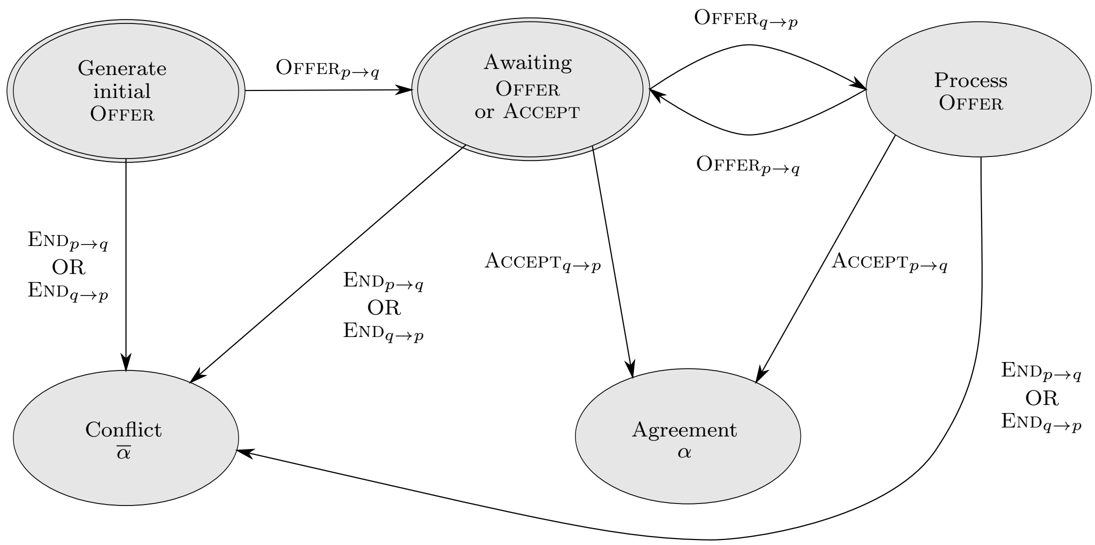


FIGURE 3.1: State diagram showing the negotiation protocol, from the perspective of a single agent, p.

the most recent OFFER it received, it can send an ACCEPT message in order to form an agreement, $\alpha \in O$. If an agreement is formed before the deadline, each player, p receives the utility of that agreement, according to their utility function, U_p, the time of agreement, t, and the discounting parameter, δ_p. Conversely, if no agreement is reached by the negotiation deadline (or if the agent terminated the negotiation by sending an END message at any time), the negotiation ends in conflict, with each player receiving a utility calculated according to the utility of conflict, $u_{\overline{\alpha}}$, the time of disagreement, t, and the discounting parameter. Figure 3.1 is a state diagram from the perspective of a single agent, showing the messages exchanged and the various states of the agent.

Each agent, p, is provided with its own utility function, $U_p(\cdot,\cdot)$, which, at time $t \in T$ (where T is the range of time during which the negotiation takes place, from the start to the deadline) maps all possible outcomes, ω in the outcome space $O \cup \{\overline{\alpha}\}$, to a value in the range $[0, 1]$. Formally:

$$U_p : (O \cup \{\overline{\alpha}\}) \times T \to [0, 1] \tag{3.1}$$

As is common in the literature, this utility function is modelled by separate components: the function $U_p(\cdot)$, which calculates the undiscounted utility of an outcome, and another, $D(\cdot,\cdot)$, which discounts that utility depending on the time that the outcome was reached and the discounting parameter. Formally:

$$U_p(\omega, t) = D(U_p(\omega), t) \tag{3.2}$$

Furthermore, the undiscounted utility of an outcome depends on whether the outcome is an agreement, α, or conflict, $\overline{\alpha}$. In the case of agreements, we define the function $U_p(\cdot)$ as being additive over all of the negotiation issues. Therefore, the undiscounted utility of agreement $\alpha \in O$ is given by:

$$U_p(\alpha) = \sum_{i=1}^{n} w_{p,i} \cdot U_{p,i}(v_i) \tag{3.3}$$

where $w_{p,i}$ is the weight of issue i to agent p and $U_{p,i}(v_i)$ is the utility to agent p of value v_i for issue i. Furthermore, without loss of generality, U_p is normalised such that the agent's best outcome has a utility of 1. Formally:

$$\exists \alpha \in O, U_p(\alpha) = 1 \tag{3.4}$$

On the other hand, the undiscounted utility of conflict is a constant, denoted $U_p(\overline{\alpha})$, which for convenience we denote as $u_{\overline{\alpha}}$.

Having considered how undiscounted utility is calculated, we now formalise the second component of our overall utility function, which considers time discounting. The time constraints considered in this work are based on the amount of real-time which has elapsed. For example, we consider negotiations which last 180 seconds (3 minutes). We normalise our representation of time, such that $t = 0$ represents the start of the negotiation and $t = 1$ represents the deadline at 180 seconds (which is the latest possible time at which an agreement can be reached). The discounted utility of an outcome (either agreement or conflict) with utility u at time t is then given by:

$$D(u, t) = \delta_p^t \cdot u \tag{3.5}$$

where δ_p is the discounting factor of agent p.

Having discussed the negotiation protocol and utility function used in the negotiations we consider, we now introduce the strategies that we have developed. The overall procedure used by all of the strategies discussed in this chapter can be described at a high level by Algorithm 1. Here, the approach is split into two main parts, to reduce the complexity of the task. The first is to develop a concession strategy to select the level of utility, u_τ, at which we generate offers at the current time, t_c. This is represented by the SetAspirationLevel function and we describe this part of the algorithm in more detail in Section 3.2. In order to maximise the utility achieved, the aspiration level must be set in a way that balances taking a tough strategy which may take a long time to reach agreement and might even result in no agreement, and conceding too quickly, giving the opponent an advantage.

Furthermore, in a multi-issue negotiation, there may be a number of different offers which have the same utility for a given agent but which offer a range of different utilities to the other agent. Specifically, for a given level of our own utility, if several offers achieve that utility, the one which maximises the utility for the opponent should be selected, as it has the highest chance of acceptance. Such an agreement is also considered to be more efficient, since neither party can unilaterally increase their utility by a large amount. This part of our strategy is represented by the GenerateOffer function, which selects one of the offers at our aspiration level, u_τ. We discuss our approach to this part of the problem in Section 3.3.

In addition to these two major components, some of the other functions in Algorithm 1 also form part of the agents' strategies. Specifically, the ConflictBest function determines when it is appropriate for the agent to end a negotiation before the deadline, and we discuss this aspect in Section 3.4. Furthermore, the AdjustUtilityPareto and AdjustUtilityConflict functions are how the agent can impose a spiteful behaviour, and we discuss these functions further in Section 3.5.

The rest of the functions included in Algorithm 1 are defined by the negotiation protocol (SendMessage, ReceiveMessage and GetOffer) or the agent's utility function (GetUtility). Specifically, the GetUtility function is equivalent to $U_p(o_{\text{opp}})$, as defined in Equation 3.3, the AdjustUtilityPareto function is equivalent to $D(S(u_{\text{opp}}), t_{\text{c}})$, as defined in Equations 3.5 and 3.24 and the AdjustUtilityConflict function is equivalent to $D(S(u_{\overline{\alpha}}, u_{\overline{\alpha}}), t_{\text{c}})$, as defined in Equations 3.5 and 3.22.

3.2 Setting the Aspiration Level

This section describes the novel concession strategy we have developed for setting our aspiration level. This is defined as the utility level at which offers are generated at a certain point in the negotiation. Furthermore, offers received from the opponent which have a utility greater than this value will be accepted. Our approach for choosing this level is to first learn the opponent's concession strategy, and then use this information to set our level as a best response to the opponent's behaviour (by adjusting our behaviour to maximise our utility given that the opponent's behaviour is fixed).

In more detail, we need to try to predict how the utility (to our agent) of the opponent's offers will vary over the duration of the negotiation. Our agent can then use this prediction to determine the best concession rate and therefore select a utility level at which to propose offers at the current time.

The general approach used in this phase of our strategies is given in Algorithm 2. Following each offer, o_{opp}, received from the opponent, the algorithm records relevant information about the offer. Then, if the input to the regression process has changed, the regression is repeated, in order to update our estimate of the future concession of the opponent. Finally, a target utility is calculated, as a best response to the learnt information.

In the remainder of this section, we consider two different approaches to predicting the opponent's behaviour. The first is to use a relatively simple, least squares regression approach, which is very fast and can therefore be repeated regularly as new offers from the opponent are observed, as we will discuss in Section 3.2.1. However, it assumes that the opponent's concession function can be fitted to a power law curve and it provides only a prediction of the opponent's future concession, without a measure of confidence in that prediction. The alternative, more advanced approach, which removes these disadvantages, is to use Gaussian process regression (described in Section 3.2.2), since this provides a confidence measure as part of its prediction. We now consider each of our approaches in turn.

3.2.1 Using Least Squares Regression

We now describe, in detail, the way in which our strategy sets its aspiration level using a least squares regression approach in order to predict the future concession of the opponent. We do this by defining the functions used in Algorithm 2.

RECORDOFFER: We assume that the function we are trying to predict, which represents the utility of the opponent's offers, according to our utility function, is non-decreasing. The rationale for this assumption is as follows. We assume that our opponent is likely to accept any offer that it has previously made, if we were to propose that offer again (since we know that any such offer gives a utility to the opponent which is high enough, otherwise it would not have proposed it initially). Due to this assumption, we record, after each offer, the current time, and the highest utility for our opponent that has been observed until that point in the negotiation. The effect of this is that, if the opponent makes an offer that is worse than an offer we have received before, this will be viewed as a lack of concession, rather than an attempt by the opponent to decrease the utility it offers to us.

RegressionRequired: Since the least squares regression can be performed in constant time, and it is also a fast approach, it is able to be repeated frequently. Therefore, we repeat the regression each time we receive a new observation.

PerformRegression: In order to approximate the opponent's concession at any future point in time during the game, we use least squares regression and in doing so assume that the observed points will roughly fit to a curve which is non-decreasing (due to the assumption that we introduced as part of the RecordOffer function), passes through the point $(0, U_p(o_{q,0}))$ (representing the initial offer made by the opponent), and does not exceed $U_p(t) = 1$ in the range $t \in [0, 1]$ (since, according to Definition 3.1, utility must be in the range $[0, 1]$). Furthermore, due to the approximate concavity of the Pareto frontier in negotiations where additive utility functions are used (as in most of the scenarios we consider), we expect the utility offered to increase more rapidly towards the start of the negotiation. Therefore, we choose the power law curve, which meets many of these requirements. Formally:

$$U_{p,\text{opp}}(t) = U_p(o_{q,0}) + e^a \cdot t^b \tag{3.6}$$

where $U_p(o_{q,0})$ is the utility, to our agent, p, of the offer made by the opponent, q, at time 0. The constants a and b are to be found by our regression approach, as discussed below.

In order to prevent the function from exceeding $U(t) = 1$ in the range $t \in [0, 1]$ (which would be unrealistic, since the utility cannot exceed 1), we use a simple heuristic to help to improve the prediction. Specifically, for each observation, we add an additional point at $(1, 0.95)$. The utility value (0.95) is close to, but below 1, in order to reduce the likelihood that the function will exceed 1.

A common approach for finding the constants a and b (in Equation 3.6), which we use as part of our strategy, is the *least-squares* curve fitting algorithm. The aim of this algorithm is to minimise the sum of the squared offsets between the data points and the fitted curve. In more detail, we find the coefficients a and b as follows (based on the equations in Weisstein (1999)):

$$b = \frac{n \cdot \sum_{i=1}^{n} \Big(\ln(t_i) \cdot \ln\big(U_i - U(o_{q,0})\big)\Big) - \sum_{i=1}^{n} \big(\ln(t_i)\big) \cdot \sum_{i=1}^{n} \Big(\ln\big(U_i - U(o_{q,0})\big)\Big)}{n \cdot \sum_{i=1}^{n} \big(\ln(t_i)^2\big) - \sum_{i=1}^{n} \big(\ln(t_i)\big)^2} \tag{3.7}$$

$$a = \frac{\sum_{i=1}^{n} \big(\ln(U_i - u_0)\big) - b \cdot \sum_{i=1}^{n} \big(\ln(t_i)\big)}{n} \tag{3.8}$$

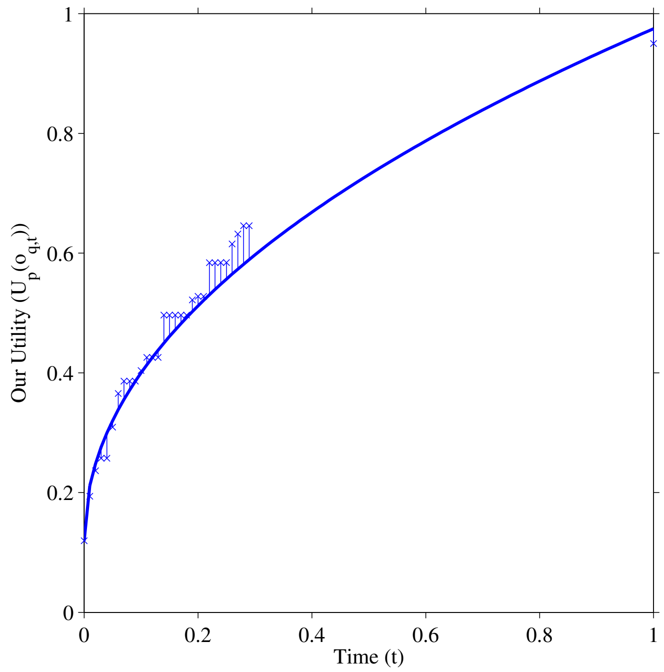


FIGURE 3.2: Estimated opponent concession. The points represent the best offers made by the opponent at that point in the negotiation, and the additional fitting points at $[1, 0.95]$. The curve is fitted through those points by minimising the sum of the squared offsets (offsets are shown as vertical lines).

where n is the number of observations of the opponent's offers, with U_i and t_i being the utility (to our agent) and time of the offer, respectively. Furthermore, $U(o_{q,0})$ is the utility of the offer received at time 0.

As an example, Figure 3.2 shows a number of observed points, and in addition shows the fitted curve (in the form of Equation 3.6), with a and b being found by Equations 3.7 and 3.8 respectively.

Once the regression coefficients, a and b, have been found, we can use Equation 3.6 to estimate the utility of the opponent's offers at any time during the negotiation session. We then have an approximation of our opponent's concession in terms of our own utility and can use this information to set our own rate of concession as a best response to this approximation.

We perform this by firstly applying our discounting function (Equation 3.5) to our model of the opponent's concession function (Equation 3.6), to create a function which gives us the discounted utility that we can expect from our opponent's offers at any point in the negotiation session. The discounted utility function is given by:

$$EU_{\text{rec}}(t) = D\left(U_{p,\text{opp}}(t), t\right) \tag{3.9}$$

GetTarget: In order to set our aspiration level as a best response to our opponent's concession, our aim is to obtain the highest discounted utility offered by the opponent (in terms of our utility). To do this, we find the maximum on the discounted opponent concession curve (given by Equation 3.9) within the time period when an agreement can be reached. By this, we mean that we ignore the parts of the curve that represent times in the past, or times which are beyond the negotiation deadline. Therefore, the next step is to solve:

$$t^* = \arg\max_{t_c \leq t \leq 1} EU_{\text{rec}}(t) \tag{3.10}$$

where t_c is the current time.

By solving Equation 3.10, our agent has identified the time t^* at which the discounted utility to our agent of our opponent's offers is likely to be maximised. The aspiration level, u^* at that time that matches the estimated utility of the opponent's offer (without any discounting) can then be found as follows:

$$u^* = U_{p,\text{opp}}(t^*) \tag{3.11}$$

We now have a point in time, t^*, at which we expect to reach agreement, and an aspiration level, u^*, that we should use at that time. Our strategy does not simply delay until t^* before conceding. Instead, it uses the intervening time to try and get an even better offer by setting the utility level above u^*, and then conceding towards u^*. Our approach to this is to use a function which passes through $[0, 1]$ (since the value of our offer at time 0 is 1), and passes through the solution to Equation 3.10 (at $[t^*, u^*]$). For an agent which aims to beat its opponent (by reaching an agreement in which it achieves a higher utility than the opponent), it is important not to reach an agreement with a low utility, as discussed in Section 2.1.9. In an attempt to avoid such agreements, we set a lower limit to the concession function, which we refer to as our reservation utility, U_{min}. We discuss how we set U_{min} in Section 3.6. Furthermore, the function should be non-increasing. Consequently, we use Faratin et al.'s time dependent concession function, which is a common approach (see Section 2.3.2.1) to finding a target utility, u_τ at time t_c, defined formally as:

$$u_\tau = 1 - (1 - U_{\text{min}}) \cdot t_c^{1/\beta} \tag{3.12}$$

where U_{min} is our reservation utility and β is the concession parameter which we choose such that the concession curve meets these constraints. Formally, β is calculated as follows:

$$\beta = \frac{\log(t^*)}{\log\left(\dfrac{1 - u^*}{1 - U_{\text{min}}}\right)} \tag{3.13}$$

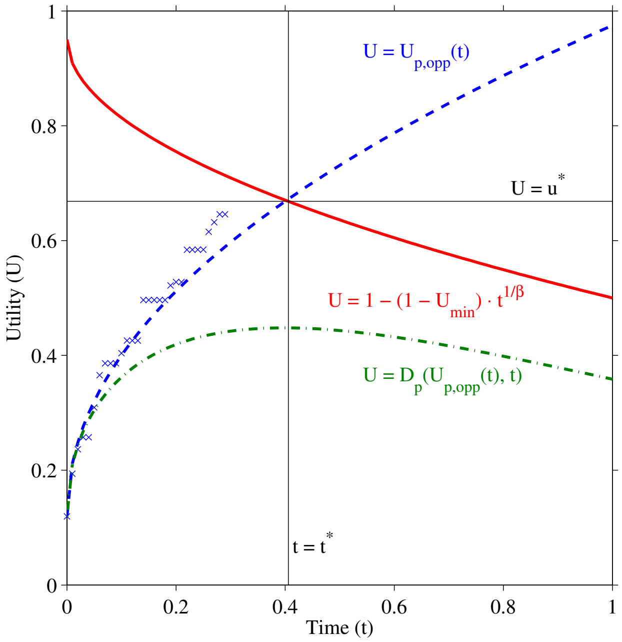


FIGURE 3.3: Example of setting our concession rate. The crosses represent the best offers made by the opponent. The dashed blue line shows the estimated future concession of the opponent (undiscounted). The dot-dashed green line shows the future discounted concession. The solid red curve is our concession curve.

Figure 3.3 shows a graphical representation of this approach. The crosses represent the best offers made by the opponent. The dashed line is a curve that is fitted through those points in order to estimate the future concession of the opponent. By applying time discounting, the dot-dashed line is produced. The maximum on the dot-dashed line is indicated by the vertical line, which represents the time, t^* at which the maximum expected discounted utility of the opponent's offers is expected to be reached. The horizontal line represents the utility, u^*, we expect from the opponent's offers at that time. The solid curve is then our curve, which passes through $[t^*, u^*]$.

Since we repeat this process following each offer, the value of β changes over time, unlike in Faratin et al.'s work where β is a fixed value. However, a possible disadvantage of this approach is that, at the beginning of the session, the curve fitting is performed through a small number of points, and therefore the curve may not accurately reflect the future concession of the opponent. If our agent learns a curve that is inaccurate, it may set its concession parameter to be too extreme, resulting in either very concessive or very tough behaviour. There are two ways in which we limit this problem. We use β' to denote the actual concession parameter that we use in selecting our target, u_τ. Firstly, we set upper and lower bounds on our actual concession parameter (β') so that the agent does not use a concession strategy that is too extreme, either by conceding too quickly at the beginning of the session, therefore reaching agreement at a low utility level, or by playing too tough and therefore conceding too late. Secondly, at the start of the game ($0 \leq t_c < 0.1$), whilst the number of observed offers is low, we use linear concession. Following this ($0.1 \leq t_c < 0.2$) we gradually increase the effect of our learnt β', and beyond this time ($0.2 \leq t_c < 1$), we use the learnt value entirely. By combining these two adjustments, our time dependent concession function becomes:

$$\beta' = \max\left(\beta_{\min}, \min\left(\beta_{\max}, \beta\right)\right) \tag{3.14}$$

$$u_\tau = U_0 - (U_{\min} - U_0) \cdot \begin{cases} t_c & \text{if } 0 \leq t_c < 0.1 \\ \left(\dfrac{t_c(t_c - 0.1) + t_c^{1/\beta'}(0.2 - t_c)}{0.1}\right) & \text{if } 0.1 \leq t_c < 0.2 \\ t_c^{1/\beta'} & \text{otherwise} \end{cases} \tag{3.15}$$

where t_c is the current time, $\beta_{\min}$ is the minimum value for β' and $\beta_{\max}$ is the maximum value for β'. In our agent, we set $\beta_{\min} = 0.01$ and $\beta_{\max} = 2.0$. The choice of these particular values were somewhat arbitrary, although we wanted to ensure that they were far enough apart to allow our strategy to be reasonably flexible, whilst avoiding the extreme behaviour discussed above. We did not attempt to optimise these parameters.

3.2.2 Using Gaussian Process Regression

In this section, we describe a more sophisticated concession process, which uses Gaussian process regression to estimate the future concession of our opponent. We use this regression technique as it provides both a prediction (of the opponent's future behaviour) and a measure of the level of confidence in that prediction. Our strategy uses this confidence measure in calculating the expected utility of future offers from the opponent, and of the offers made by our agent. If the confidence is low, it may be necessary for our strategy to concede more in order for its offers to have a reasonably high probability of acceptance.

Again, we consider this approach in terms of the functions used in Algorithm 2.

RecordOffer: As input to the Gaussian process, we use the maximum value offered by the opponent in a particular time window of duration t_{window}, and the time of that window. The reason for using this windowed approach is twofold. Firstly, it reduces the effect of noise on the Gaussian process. Since we measure the utility of the opponent's offers in terms of our agent's utility function, it is possible that this value may vary significantly in a given window. This is due to the offers consisting of multiple issues, with the negotiation partners having different utility functions. In such an environment, it is possible that a small change in utility for the opponent can be observed as a large change by our agent. We use the maximum value in each time window, rather than the average, as the maximum represents the best offer that we have observed, and can therefore expect to reach agreement at. Secondly, it reduces the amount of input data for the Gaussian process. If all of the observed offers were used, there could be thousands of data points, which could significantly slow down the regression process and therefore delay the negotiation.

RegressionRequired: Since the input to the regression only changes at the end of each time window, there is no benefit to repeating the regression within a single time window. Therefore, using this strategy, the regression is only performed if the current time window is different to that when the previous regression was performed.

PerformRegression: Our agent uses a Matérn covariance function and a linear mean function (Rasmussen and Williams, 2006). The Matérn covariance function is a stationary function. That is, it is based only on the distance between two points. Furthermore it is a decreasing function, such that the covariance between two points decreases as the distance between them increases. Given that we have little other information about the expected behaviour of the opponent, we consider the Matérn covariance function to be the most appropriate for our work.[1] We selected a linear mean function as we expect the offers of the opponent to increase over time. Whilst this increase may be non-linear, the linear mean is a simple approximation, which is much more appropriate than a constant mean. Furthermore, by using a Matérn covariance function and a linear mean function, the regression is fast enough to be computed in real time during the negotiation.

The output of the Gaussian process is a Gaussian probability density function, for each time t, of the form:

$$f(u; \mu_t, \sigma_t) = \frac{1}{\sqrt{2\pi\sigma_t^2}} e^{-\dfrac{u-\mu_t}{2\sigma_t^2}} \tag{3.16}$$

where μ_t and σ_t are the mean and standard deviation, respectively. The mean, μ_t, gives an indication of the expected value for u at time t, whilst the standard deviation, σ_t, is an indication of how accurate the prediction of μ_t is likely to be.

We note that alternative regression techniques can be used in place of a Gaussian process, such as Bayesian linear regression, providing their output contains both mean and standard deviation measures. However, in this work, we have only evaluated our approach using a Gaussian process.

Figure 3.4 shows an example of the input to and output from the Gaussian process performed at time $t_c = 0.25$ during a negotiation with *Agent K* in the *Itex vs Cypress* scenario (see Section 4.3 for more regarding the scenarios and opponents).

[1] Alternative stationary covariance functions include the exponential and squared exponential covariance functions, both of which are special cases of the Matérn covariance function.

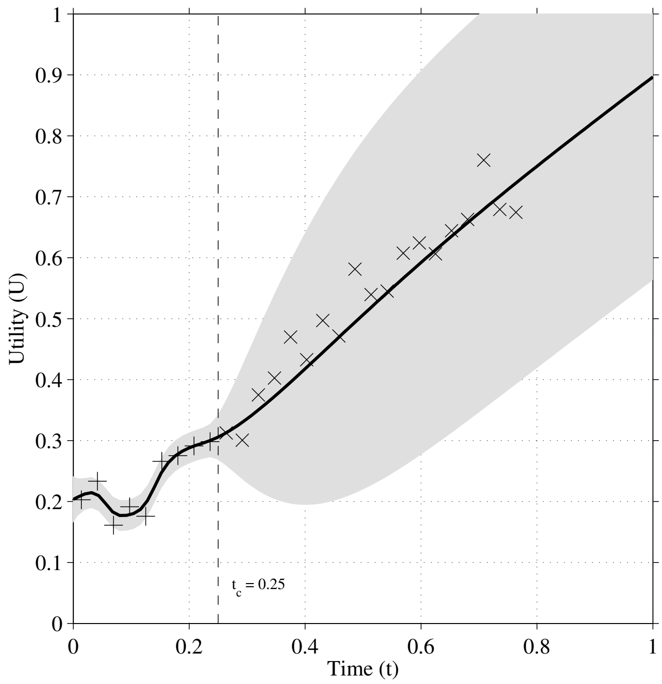


FIGURE 3.4: Demonstration of the Gaussian process used in a negotiation with *Agent K* in the *Itex vs Cypress* scenario (taken from ANAC2010, see Section 4.3 for further details), at time $t_c = 0.25$. The plus signs are the input data, based on the observed offers. The crosses are based on future offers. The mean of the Gaussian output is shown as a solid line, with the shaded area representing the 95% confidence interval.

As given by Definition 3.1, we assume that the utility of the opponent's offers must lie in the range $[0, 1]$. Therefore, we adjust the output of the Gaussian process, to create a truncated normal distribution, constrained to fit in the utility range $[0, 1]$, as follows:

$$p_{[0,1]}(u; \mu_t, \sigma_t) = \frac{p(u; \mu_t, \sigma_t)}{P(1; \mu_t, \sigma_t) - P(0; \mu_t, \sigma_t)} \tag{3.17}$$

where the mean, μ_t, and variance, σ_t, are those given by the Gaussian process, $p(u; \mu_t, \sigma_t)$ is as given in Equation 3.16 and $P(u; \mu_t, \sigma_t)$ is the cumulative distribution for $p(u; \mu_t, \sigma_t)$. That is:

$$P(u; \mu_t, \sigma_t) = \int_0^u p(x; \mu_t, \sigma_t) \mathrm{d}x \tag{3.18}$$

Based on the prediction of the opponent's future concession which was generated using the regression technique, our strategy then aims to set its concession by optimising the expected utility given that prediction.

GETTARGET: Having introduced our use of Gaussian processes in predicting the future concession of the opponent, we now discuss the main contribution of this work to the literature, which is to show how the output of a Gaussian process can be used in setting the concession rate. Specifically, our approach is the first practical concession strategy for multi-issue negotiation with real-time constraints to use both the mean, μ, and standard deviation, σ, output by the Gaussian process in setting an optimal concession rate. The aim of this stage of our strategy is to calculate the best time, t^*, and utility value, u^*, at which to reach agreement. To reduce the complexity of this part of the problem, we use a heuristic which first finds t^* and then uses it to calculate u^*. We therefore consider the best time, t^*, to be the point in future time ($t \in [t_c, 1]$) at which the expected utility of the opponent's offers is maximised, using:

$$t^* = \underset{t \in [t_c, 1]}{\arg\max}\, EU_{\text{rec}}(t) \tag{3.19}$$

where t_c is the current time and $EU_{\text{rec}}(t)$ is the expected utility, adjusted by the agent's spitefulness (which we introduced in Section 2.1.9) and discounting, of reaching an agreement at time t, given by:

$$EU_{\text{rec}}(t) = \int_0^1 p_{[0,1]}(u; \mu_t, \sigma_t) D\left(S(u), t\right) \mathrm{d}u \tag{3.20}$$

where $D(\cdot, \cdot)$ is the discounting function, given by Equation 3.5, $S(\cdot)$ is the spitefulness function (which we will discuss in Section 3.5), and $p_{[0,1]}(\cdot)$ is the probability distribution over the values of u, as determined by our regression process.

Having selected the time, t^*, at which the expected utility of the opponent's offers is maximised, our agent needs to choose a utility, u^*, to offer at that time. The approach that our strategy takes here is to maximise the expected utility of making an offer of utility u. We assume that an offer of utility u will be accepted at time t^* if $u \leq u_{t^*}$. Since we have a probability distribution over u_{t^*}, we can calculate the probability that $u \leq u_{t^*}$ using the truncated cumulative distribution $P_{[0,1]}(u; \mu_t, \sigma_t)$. Therefore, the utility, u^*, which should be offered at time t^*, is given by:

$$u^* = \underset{u \in [0,1]}{\arg\max}\, P_{[0,1]}(u; \mu_{t^*}, \sigma_{t^*}) D\left(S(u), t^*\right) \tag{3.21}$$

where $D(\cdot, \cdot)$ and $S(\cdot)$ are as before, and $P_{[0,1]}(\cdot)$ is the cumulative distribution for $p_{[0,1]}(\cdot)$.

Finally, having determined u^* as the utility to offer at time t^*, our agent needs to choose a utility to offer at the current time, t_c. The approach used here is the same as in Section 3.2.1 (Equations 3.13, 3.14 and 3.15).

Having shown our concession strategy, which considers the opponent's concession, the discounting factor, the deadline and our reservation utility, we now discuss the other major aspect to our agent, which is the strategy it uses for choosing the value of each individual issue.

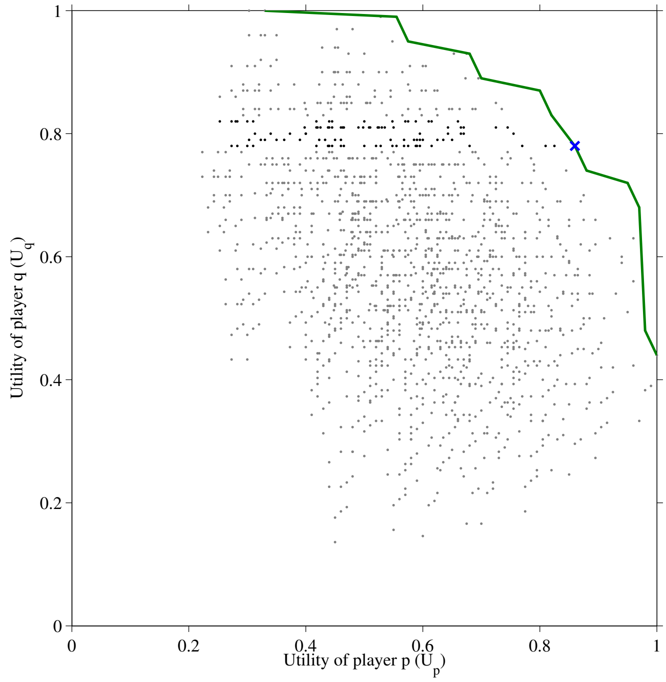


FIGURE 3.5: Outcome space for an example scenario. The set of offers which give agent q a utility of 0.8 ± 0.025 are displayed in black. The Pareto frontier is displayed as a solid line, and the Pareto-efficient offer with a utility to agent q of 0.78 is marked with a cross.

3.3 Negotiating over Multiple Issues

In the previous section we showed how our agent can choose its aspiration level at any point during the negotiation session. However, in a multi-issue negotiation, there are likely to be a number of different offers at any given utility level. Our agent is indifferent between this set of offers, since they all result in the same utility from its perspective. However, the opponent is unlikely to be indifferent between the offers, since its utility function is likely to be different to ours. Our aim and basis of our negotiation strategy is to select the offer (from the set of offers over which our agent is indifferent) which maximises the utility of the opponent. The reason for doing so is that the opponent is more likely to accept offers with a higher utility. In addition, from a performance perspective, such outcomes are closer to Pareto-efficient. To illustrate, Figure 3.5 shows an outcome space in grey, with the set of offers with a utility for our agent of 0.8 displayed in black. The Pareto frontier is displayed as a solid line, and the Pareto-efficient offer with a utility to our agent of 0.8 is marked with a cross.

In the remainder of this section, we present our approach to selecting an offer at a given utility level. Firstly, we discuss the basic *random selection* approach (Section 3.3.1), before presenting an additional technique which can be used in conjunction with this approach in order to enhance the opportunity for an agreement (Section 3.3.2).

3.3.1 Random Selection

With software agents (which have fast reaction times in the order of milliseconds), under real-time constraints, the goal is to reach an agreement within a short time period, but not necessarily to limit the number of offers made. Therefore, if our agent can generate an offer quickly (even if it does so at random), it can explore more of the outcome space in the available time. A fast method for selecting a package is to do so at random. Therefore, we begin by considering such an approach.

Ideally, the aim of our random selection strategy (and the other selection strategies we discuss in this Section) is to select an offer with utility u_τ. However, it may be difficult, or in a discrete domain, impossible to find such an offer. Consequently, our random selection strategy chooses an offer which has a utility close to the target, u_τ, by generating a random offer with a utility in the range $[u_\tau - 0.025, u_\tau + 0.025]$. If an offer cannot be found within this range, the range is expanded, until a solution is found. To avoid selecting an offer that is lower than our initial lower limit where there are potential offers that lie above our initial upper limit, the search range is firstly incrementally expanded upwards, continuing to exclude offers with utilities lower than $u_\tau - 0.025$. If an offer still cannot be found once the upper search limit has reached 1, the range is then incrementally expanded downwards. While this approach to selecting an offer is a simple one it has produced very good results. In particular, when combined with a good regression method, the results are often better than for more advanced approaches, as we will show in Chapter 4. Furthermore, random selection is a very good benchmark strategy, which can be used to compare against more advanced strategies.[2]

3.3.2 Re-proposal of Best Offer

Using the random approach that we have just introduced, our strategy can propose a large number of offers in a limited time period. However, in certain circumstances, there may be an easily identified offer at a given utility level which is expected to be accepted by the opponent. Therefore we now discuss an additional feature that can be used in conjunction with this approach in order to improve the likelihood that our offers are accepted, without reducing our aspiration level.

This feature works as follows. If the aspiration level, u_τ (as determined by one of the strategies introduced in Section 3.2) drops below the highest utility offered by the opponent so far during the negotiation, instead of proposing a random offer according

[2] A variant of this approach is weighted random selection, in which a range of possible offers are generated, and evaluated according to an estimate of the opponent's utility function. An offer is then selected at random from this set, weighted by this estimate.

to the strategy discussed earlier in the section, we propose the best offer (according to our utility) that we have received from the opponent. The reason for proposing this offer is as follows. Its utility is at least u_τ, so from our agent's perspective, it is no worse than any other offer with utility u_τ. If the opponent is using a non-increasing time-dependent concession function (such as the one we introduce in Equation 3.12), then the offer it proposed at time t will be accepted at any later time, $t' > t$. Therefore, assuming that the opponent is using such a concession function, we can be sure that the offer will be accepted, whereas any other offer with utility u_τ may not be. If the offer is not accepted, we continue to make offers according to the approaches introduced earlier in this section, (until the best offer has changed, when we will again consider proposing it using this feature).

3.4 Handling Non-Zero Utility of Conflict

In negotiations where the utility of conflict is non-zero, it is necessary to consider the possibility that the conflict outcome may be more desirable than some agreements. A simple approach to address such situations is to avoid making or accepting offers which have a utility lower than the utility of conflict. However, in a negotiation where time discounting is also present, such an approach may not be sufficient. In more detail, time discounting is applied according to the time at which the outcome (either agreement or conflict) is reached. Therefore, there may be a benefit to ending a negotiation early, in order to avoid the discounting factor having a significant effect.

The approach taken by our strategy is to compare the value of conflict ($u_{\overline{\alpha}}$) at the current time (t_c), with the value of the best offer (u^*) expected to be available in the future, at that time (t^*). If $u_{\overline{\alpha}} \cdot \delta^{t_c} > u^* \cdot \delta^{t^*}$, the agent will terminate the negotiation (obtaining a utility of $u_{\overline{\alpha}} \cdot \delta^{t_c}$), otherwise, negotiations will continue.

In scenarios where there is no discounting ($\delta = 1$) or where the utility of conflict is zero ($u_{\overline{\alpha}}$), the value of conflict at the deadline is equal to the value of conflict at any other time during the negotiation. Therefore, in such scenarios, there is no benefit to breaking off a negotiation prior to the deadline (note that if no agreement is reached by that time, the default outcome is conflict).

3.5 Spitefulness

In some settings, such as in a tournament, the goal may not be for an agent to maximise its own utility but, rather, to beat any opponents. That is, to obtain a utility higher than the opponent's. In such a setting, the agent can no longer take a purely self-interested approach, as it may be able to benefit from harming the performance of its opponent. We refer to such behaviour as *spitefulness*, and we now discuss the spitefulness function, $S(u)$, in more detail. The aim of the spitefulness function is to encourage our strategy to reach agreements in which our player, p, achieves a higher score than that of its opponent, q.

We define a generic spitefulness function of the form

$$S(u_p, u_q) = s + (1 - s) \cdot u_p - s \cdot u_q \tag{3.22}$$

where u_p and u_q are the utilities of our player and its opponent, and $0 \leq s \leq 1$ is the spitefulness parameter. However, since the utility, u_q, of our opponent cannot be observed, we require a spitefulness function which does not depend on it.

To generate such a spitefulness function, we assume that an increase in our utility, u_p, is likely to lead to a decrease in our opponent's utility, u_q. In a multi-issue negotiation with an additive utility function (as in the negotiations we consider), the Pareto frontier is convex. As a result, with the assumption that the offers being made are Pareto efficient, a small decrease in u_p, for high values of u_p, results in a large increase in u_q. At lower values of u_p, a small decrease in U_p has little effect on u_q. We can therefore assume that, at high values of u_p, the adjusted utility for a spiteful strategy should be higher, but change more rapidly than for low values of u_p.

In order to estimate u_q given u_p, we assume that the Pareto frontier can be approximated by a curve, of the form:

$$u_p^k + u_q^k = 1 \tag{3.23}$$

where $k \geq 1$ is the competitiveness coefficient.

In order to estimate k, we first assume that the opponent concedes at the same rate as our strategy. Under this assumption, the opponent's utility, u_q, of the offer made by the opponent at a given time is equal to our utility, u_p, of the offer our strategy makes at the same time. We then find the minimum value that k can take, such that all offers made so far lie beneath the approximated Pareto frontier.

If we then consider the effect of concession, assuming that all offers made are Pareto efficient, we find the spitefulness function, $S(\cdot)$ to be given by:

$$S(u_p) = s + (1 - s) \cdot u_p - s \cdot (1 - u_p^k)^{1/k} \tag{3.24}$$

where $k \geq 1$ is the competitiveness coefficient and $0 \leq s \leq 1$ is the spitefulness parameter.

A spite level of zero ($s = 0$) represents non-spiteful behaviour, that is, the agent is only interested in maximising its utility (and in this case, $S(u_p) = u_p$). In contrast, for spite levels greater than zero ($s > 0$), the agent regards lower utility agreements with much lower value than their true utility.

When combined with our Gaussian process regression, spitefulness has an important effect on our strategy's calculation of expected utility. In more detail, the effect of the standard deviation, σ_t (in Equation 3.20) on a spiteful agent is as follows. If, at two points in time, t_1 and t_2, the mean values are the same ($\mu_{t_1} = \mu_{t_2}$), but the standard deviation differs such that $\sigma_{t_1} < \sigma_{t_2}$, then a spiteful agent will consider the expectation at time t_2 to be greater than at time t_1. That is, the spiteful agent is prepared to wait for the less certain offer at time t_2, as there is a higher chance that the utility may significantly differ from the value of μ_{t_2}, than it would for μ_{t_1}. Of course, that difference may be positive or negative, but, for a spiteful agent (where $s > 0$), $S(\mu + x) + S(\mu - x) > S(\mu) + S(\mu)$ (where $x > 0$ is the difference from the mean, μ). In contrast, non-spiteful agents are indifferent between the two solutions (since, if $s = 0$, $S(\mu + x) + S(\mu - x) = S(\mu) + S(\mu)$).

3.6 Reservation Utility

In order to prevent the strategy from conceding too much, perhaps against a non-concessive opponent, a reservation utility can be used. The reservation utility is the minimum utility level that our strategy will concede to. Since our utility function is normalised such that all outcomes have a utility in the range $[0, 1]$, a simple approach is to use a static reservation utility, with a value of $U_{\min} = 0.5$.

A more advanced approach is to determine the reservation value as a function of the scenario. The aim is to choose a value which is large enough that it reduces the risk of our agent being exploited by a tough, non-concessive opponent, but at the same time is small enough that it allows agreements to be reached. Our approach is to choose a value, given the set of outcomes, O and our utility function U_p. If we assume that opponent q's preferences are strictly opposed to those of our agent, p (that is, $\forall o, o' \in O, U_p(o) > U_p(o') \Leftrightarrow U_q(o) \leq U_q(o')$) and that the opponent chooses its reservation value using the same approach, then the highest reservation value which we can choose is equal to the median value of $U_p(o)$. If either party chooses a higher value, it is possible that there are no outcomes that are mutually acceptable.

In order to determine this median value, it is first necessary to calculate $U_p(o)$ for all outcomes, $o \in O$. In a scenario with a large outcome space, this may require significant time and memory. Therefore, in practice we use an approximation of the median, by taking a random sample of $U_p(o)$, gathering the utilities of as many outcomes as possible within a short time period (2 seconds in our case) and taking the median of this value.

3.7 Our One-to-One Negotiation Agents

Having described the different parts of our strategies, we now introduce the agents that we have developed, which use various combinations of these strategies. All of our agents are implemented using the Java programming language, under the framework provided by the GENIUS environment (introduced in Section 2.2.1). Through the use of a common framework, our agents can be compared with each other and, furthermore, with other agents developed under the same framework. We do this in the evaluations we carried out as part of Chapter 4. Where appropriate, we have used library implementations of common functions.

3.7.1 IAMhaggler2010

IAMhaggler2010 was one of our early agents, developed for the first Automated Negotiating Agents Competition (ANAC 2010). (Note that it was initially submitted under the name 'IAMhaggler'.) It uses the least squares regression approach (described in Section 3.2.1), with a fixed reservation utility, $U_{\min} = 0.5$, combined with the Pareto-search method. (The Pareto-search method is designed for negotiation over continuous issues. We do not consider such issues in this thesis, but for completeness, the method is described in Appendix B.) Furthermore, it uses the first approach to re-proposing the opponent's best offer (described in Section 3.3.2).

3.7.2 IAMhaggler2011

Following ANAC 2010, we found that by using the random selection method (described in Section 3.3.1), our agent could make thousands of offers during a 3 minute negotiation, and was therefore able to search a large outcome space and achieve high utility agreements, without modelling the opponent's preferences. We therefore chose to focus on improving our approach to learning the opponent's concession, as this aspect showed more scope for improvement. In doing so, IAMhaggler2011 was developed and submitted to the second Automated Negotiating Agents Competition (ANAC 2011). It uses the Gaussian process regression approach (described in 3.2.2), with a fixed reservation utility, $U_{\min} = 0.5$, combined with the random selection method (described in

Section 3.3.1). The strategy's behaviour was adjusted using a spitefulness function, which was less sophisticated than the one described in Section 3.5. In more detail, the function $S(u) = u^s$ was used, with $s = 1$ (non-spiteful) in the qualifying round of the competition, and $s = 3$ (spiteful) for the final round.[3] Furthermore, it uses the second approach to re-proposing the opponent's best offer (described in Section 3.3.2).

This agent makes use of *Commons-Math: The Apache Commons Mathematics Library*[4], *JAMA, A Java Matrix Package*[5] and the *Gaussian Process Regression for Java* library[6].

3.7.3 IAMhaggler2012

To prepare for ANAC 2012, IAMhaggler2011 was developed further into IAMhaggler2012. ANAC 2012 introduced non-zero utility of conflict, so in order to handle this, we used the approach described in Section 3.4. Furthermore, the approach to spitefulness was improved, using the approach described in Section 3.5. Following the competition, we made some further improvements, and we now set our reservation utility, $U_{\min}$ depending on the scenario, using the approach described in Section 3.6.

3.7.4 IAMcrazyHaggler

Whilst considering the design of agents which adapt to their opponent's behaviour (such as the IAMhaggler agents), we decided to investigate whether a simple but tough strategy was capable of achieving high utility agreements against agents which are highly adaptive to the behaviour of their opponents. Such a strategy can be considered to be an interesting benchmark strategy against more complex opponents. Specifically, we developed the IAMcrazyHaggler series of agents, which use a very simple strategy that makes random offers which have a utility over a fixed *offer threshold* and accepts offers made by the opponent if their utility is greater than a further fixed *acceptance threshold*.

The different versions of IAMcrazyHaggler differ only in their thresholds, as we will now discuss:

[3] Note that spitefulness was referred to as risk aversion in the resulting publication, which appears in the Proceedings of the 22nd International Joint Conference on Artificial Intelligence.

[4] `http://commons.apache.org/math/`

[5] `http://math.nist.gov/javanumerics/jama/`

[6] `https://forge.ecs.soton.ac.uk/projects/gp4j/`

IAMcrazyHaggler2010 Our first version, now known as IAMcrazyHaggler2010[7], uses a very high offer threshold ($U_p(o_{p,t}) > 0.95$) and a slightly lower acceptance threshold ($U_p(o_{q,t}) > 0.90$). If the scenario contains a discounting factor, the offer and acceptance thresholds are each reduced by 0.05 in an attempt to reach agreement more quickly, and therefore reduce the loss due to discounting.

IAMcrazyHaggler2011 We found that, at ANAC 2010, IAMcrazyHaggler2010 failed to reach agreement in many negotiation sessions, due to its extremely high threshold. Therefore, for ANAC 2011 we developed IAMcrazyHaggler2011. It works in the same way as IAMcrazyHaggler2010, but with much lower thresholds ($U_p(o_{p,t}) > 0.70$, $U_p(o_{q,t}) > 0.65$). Due to these lower thresholds, IAMcrazyHaggler2011 is able to propose and accept offers from a much larger part of the outcome space. This should lead to more agreements, although individually, they are likely to achieve a lower utility for the agent. In contrast to IAMcrazyHaggler2010, this agent does not adjust its behaviour in discounted scenarios, as its threshold is already rather low.

IAMcrazyHaggler2012 At ANAC 2011, IAMcrazyHaggler2011 reached agreement in most negotiation sessions, due to it having a relatively low threshold. The scenarios chosen for ANAC 2011 tended to be much less competitive than those included in ANAC 2010, in that it was often possible for agents to reach agreements which offered high utility for both parties. As a result, IAMcrazyHaggler2011 reached agreements that were considerably lower in value than the agreements formed by many of the more advanced strategies. Therefore, for ANAC 2012, we created IAMcrazyHaggler2012 as a further variant of the IAMcrazyHaggler series of agents, this time increasing the thresholds to ($U_p(o_{p,t}) > 0.8$, $U_p(o_{q,t}) > 0.8$).

As we will show in the evaluation of our one-to-one strategies (Chapter 4), the performance of an agent with fixed thresholds is highly dependent on the value of those thresholds.

3.8 Summary

In this chapter, we have described the negotiating agents that we have designed. We detailed the two key parts of our strategies: the processes that our agents use to set their aspiration levels, and the ways in which they select an offer with a given utility. In doing so, we have combined and extended several existing approaches from the literature, and designed two new adaptive strategies.

[7]The agent was entered into ANAC 2010, under the name IAMcrazyHaggler.

Specifically, we have contributed the following to the literature on automated negotiation:

- We have developed a novel strategy, which uses both a Gaussian process prediction and the certainty of that prediction, to calculate the concession an agent should make over time. This strategy is able to negotiate directly with an unknown opponent and uses a principled approach, by firstly predicting the opponent's future behaviour and then adapting to the agent's offers in order to maximise the expected utility of agreement. Furthermore, the proposed strategy is designed to deal with real-time constraints in multi-issue negotiation where the issues have an unknown ordering.

Furthermore, against the requirements set out in Section 1.2, we have designed six agents which:

- work in a decentralised manner, communicating directly with the other negotiating agents (Requirement 1),
- are able to negotiate in an environment without knowledge of the preferences or behaviour of any other party (Requirement 3),
- have been designed to negotiate in domains with multiple issues (Requirement 4),
- are able to negotiate over discrete issues without a known, common ordering (Requirement 5), and
- support real-time constraints (Requirement 6).

However, none of these strategies are designed to coordinate concurrent negotiation with more than two parties (Requirement 8) and therefore they also do not consider decommitment (Requirement 9). We consider this aspect in Chapter 5.

In the following chapter, we will evaluate the performance of our agents, checking them against the remaining two requirements of efficiency (Requirement 2) and computational tractability (Requirement 7).

Algorithm 1 Overview of our general negotiation process, which is common to all of our strategies. Let $u_{\overline{\alpha}}$ represent the utility of conflict and let t_c represent the current time.

```
while t_c ∈ [0,1] do
  m ⇐ ReceiveMessage()
  if m is not an Offer then
    return
  end if
  o_opp ⇐ GetOffer(m)
  u_τ ⇐ SetAspirationLevel(o_opp, t_c)
  u_opp ⇐ GetUtility(o_opp)
  u_1 ⇐ AdjustUtilityConflict(u_ᾱ, t_c)
  u_2 ⇐ AdjustUtilityPareto(u_opp, t_c)
  u_3 ⇐ AdjustUtilityPareto(u_τ, t_c)
  if ConflictBest(u_1) then
    SendMessage(End)
    return
  else if  u_2 ≥ u_3 then
    SendMessage(Accept(o_opp))
    return
  else
    o_own ⇐ GenerateOffer(u_τ)
    SendMessage(Offer(o_own))
  end if
end while
```

Algorithm 2 Overview of our function for setting the aspiration level, following an offer, o_{opp}, from the opponent, at time t_c.

```
Function SetAspirationLevel(o_opp, t_c)
Require: o_opp, t_c
  RecordOffer(o_opp, t_c)
  if RegressionRequired(t_c) then
    PerformRegression()
  end if
  return  GetTarget(t_c)
```

Chapter 4

Evaluation of One-to-One Negotiation Agents

This chapter evaluates the performance of our one-to-one negotiation agents, using the techniques and against the performance criteria that we discussed in Section 2.2. Throughout this chapter, we analyse the results of negotiations from a variety of scenarios. Each scenario consists of a domain and, for each player, an associated preference profile. The domain specifies the set of issues that are being negotiated over, and the range of values that each issue can take, and therefore determines the set of outcomes that are possible. The preference profiles define each agent's utility function. Formal definitions of domains and preference profiles can be found in Section 2.2.1.

The remainder of this chapter is structured as follows. We begin by introducing the scenarios that we use in our evaluation (Section 4.1) before finding a suitable spitefulness parameter for use in tournament settings (Section 4.2). We then provide a summary of the performance of the agents in the Automated Negotiating Agents Competitions (Section 4.3), before analysing the performance of each of the agents in self play (Section 4.4). Next, we carry out a more extensive evaluation of their performance in tournaments (Section 4.5) and we also evaluate the tractability of our strategies by considering the offer rates achieved by all of the strategies (Section 4.6). Subsequently, we consider tournaments in which more than one player uses a particular strategy, by performing an empirical game theoretic analysis of the results (Section 4.7). We conclude with a summary of the chapter (Section 4.8).

4.1 Evaluation Scenarios

In this evaluation, we use the scenarios of the most recent Automated Negotiating Agents Competition (ANAC 2012). The scenarios used in ANAC 2012 came from three sources, as follows. Each of the 17 participants of ANAC 2012 submitted a scenario. To this, the organisers added the scenarios of the ANAC 2011 final (excluding any that had been resubmitted by the 2012 participants) and the scenarios of ANAC 2010 (excluding any that had been resubmitted by the 2011 or 2012 participants) to produce a total of 24 scenarios. Appendix A provides full details of the utility functions used, along with plots of the outcome spaces, for each of these scenarios.

In order to evaluate the performance of the agents in different types of scenario, we classify them according to a number of characteristics. Firstly, they are classified according to the size of the outcome space of their domain, $|O|$. We partition the scenarios into three size classes, with an equal number of scenarios in each class, as follows: small ($|O| < 200$), medium ($200 \leq |O| < 3500$) and large ($3500 \leq |O|$).

Secondly, they are classified according to their competitiveness, $C(O)$, defined as the minimum distance from a point in the outcome space to the point which represents complete satisfaction (that is, the point at which each agent achieves a utility of 1).[1] We partition our scenarios into three competitiveness classes, with an equal number of scenarios in each class, as follows: low ($C(O) \leq 0.22$), medium ($0.22 < C(O) \leq 0.30$), high ($0.30 < C(O)$).

The domain sizes and competitiveness of the scenarios can be seen in Table 4.1. The scenario with the smallest domain, *NiceOrDie*, has a single issue with just 3 possible outcomes, whereas the one with the largest, *Energy*, has 8 issues and a total of 390625 possible outcomes. The least competitive scenario, *ADG*, has a competitive value of 0.092, whereas the most competitive, *NiceOrDie*, has a competitiveness value of 0.840.

The 2012 edition of the competition was the first to include non-zero utilities of conflict. In order to evaluate the agents using a range of discounting factors and utilities of conflict, we take the following approach: Three discounting factor parameters ($\delta \in \{0.50, 0.75, 1.00\}$) and three utility of conflict parameters ($u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$) are chosen. Each of the 24 scenarios are tested with each value of δ and $u_{\overline{\alpha}}$. Therefore, once

[1] Note that the definition of competitiveness introduced here differs from that used in the context of auctions. In an auction, if there are more agents competing for the same resources, the competitiveness increases (typically leading to higher prices). In contrast, this thesis considers competitiveness between each pair of agents, which results from incompatibilities between their respective preferences. More compatible preference profiles enable more satisfying (i.e. closer to a utility of 1 for both agents) outcomes, and therefore a less competitive scenario.

Name	Year	Domain Size	Size Class	Competitiveness Value	Competitiveness Class
NiceOrDie	2011	3	small	0.840	high
Fifty fifty	2012	11	small	0.707	high
Laptop	2011	27	small	0.160	low
Flight Booking	2012	36	small	0.281	medium
Rental House	2012	60	small	0.327	high
Barter	2012	80	small	0.492	high
Outfit	2012	128	small	0.198	low
Itex vs Cypress	2010	180	small	0.431	high
Housekeeping	2012	384	medium	0.272	medium
IS BT Acquisition	2012	384	medium	0.117	low
Airport Site Selection	2012	420	medium	0.285	medium
England vs Zimbabwe	2012	576	medium	0.272	medium
Barbecue	2012	1440	medium	0.238	medium
Grocery	2011	1600	medium	0.191	low
Phone	2012	1600	medium	0.188	low
Amsterdam Party	2011	3024	medium	0.223	medium
Fitness	2012	3520	large	0.275	medium
Camera	2012	3600	large	0.218	low
Music Collection	2012	4320	large	0.150	low
ADG	2011	15625	large	0.092	low
Energy (small)	2012	15625	large	0.430	high
Supermarket	2012	98784	large	0.347	high
Travel	2010	188160	large	0.230	medium
Energy	2012	390625	large	0.525	high

Table 4.1: Scenario characteristics.

these parameters were considered to be part of the scenario, a total of $3 * 3 * 24 = 216$ scenarios are used.

4.2 Spitefulness

We begin by carrying out experiments to determine the most suitable value for the spitefulness parameter, s. For these experiments, in order to avoid tuning our strategy to the scenarios used in the rest of our evaluation (presented in Section 4.1) we used a different set of scenarios. As opponents, we used the agents from ANAC 2012.

Specifically, we created a set of variants of the *IAMhaggler2012* agent, each with a different value of $s \in \{0, \frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}\}$. For each of these variants, we ran negotiations against all 7 opponents, in all scenarios and we calculated the tournament scores for each agent.

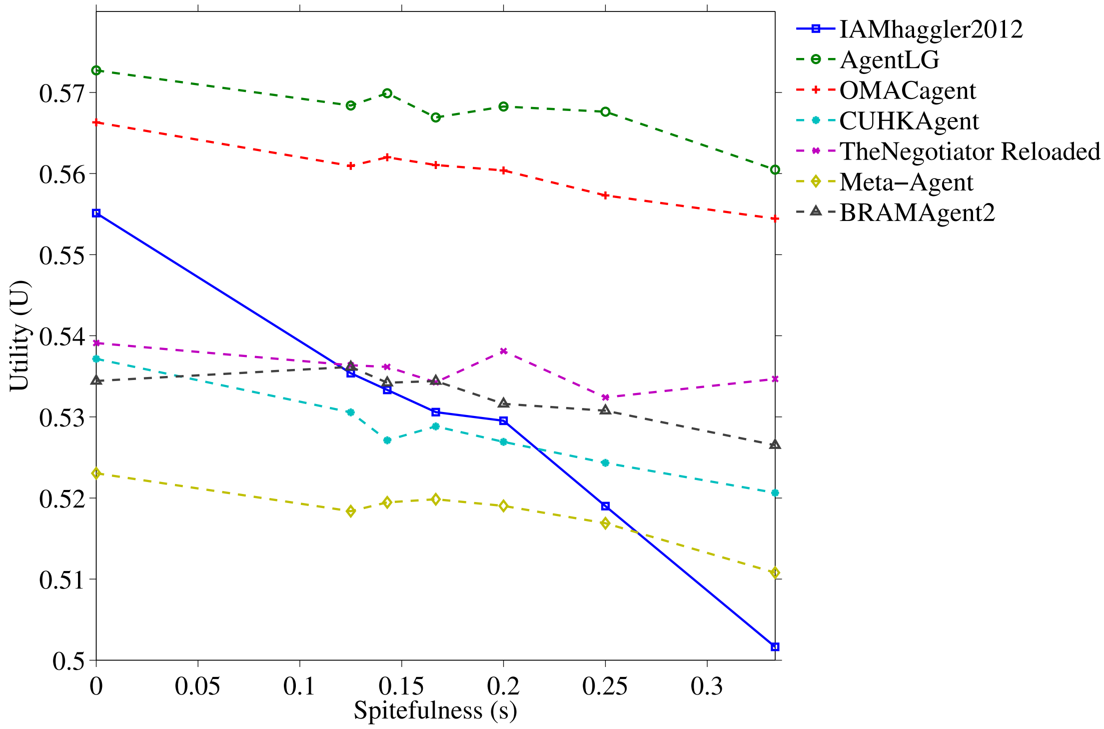


FIGURE 4.1: Average utilities achieved in 8-player tournaments, for different spitefulness values, s. *Meta-Agent* is omitted from this plot as its utility was considerably lower than that of the other agents, but showed a similar, decreasing trend to those of the opponents.

Figure 4.1 shows these tournament scores. The results reveal that, as s increases (and so our agent becomes more spiteful), the average utility of each of the opponents decreases. However, the average utility of our agent also decreases, and at a much greater rate. This is because, by being more spiteful, our agent achieves better agreements, but less often, and each time it fails to reach an agreement, a significant loss of utility is incurred by both negotiating parties. Our agent can obviously only affect the utility achieved in negotiations that it participates in. In any tournament, our agent is obviously able to have a huge effect on its score, but in a tournament setting with n players, it only participates in $\frac{1}{n-1}$ of the negotiations which affect any of its $n-1$ opponents, and therefore can only affect $\frac{1}{n-1}$ of any individual opponent's score. Therefore, by taking a more spiteful approach and failing to reach as many agreements, our agent significantly decreases its score whilst having relatively little effect on each opponent.

To confirm this, we consider the smallest possible 'tournament', with just 2 players. In such a tournament, our agent participates in all negotiations, and is therefore able to equally affect its score and that of its opponent. We create an opponent which represents all 7 ANAC 2012 opponents by averaging over the scores of all of those agents in negotiations against *IAMhaggler2012*. Figure 4.2 shows the result in this setting. In common with the larger tournament, we see the utilities of both agents being reduced

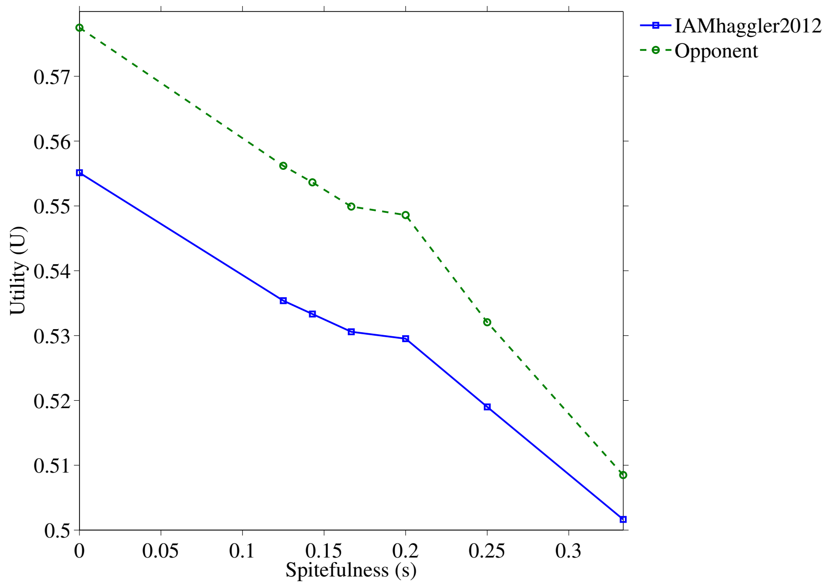


FIGURE 4.2: Average utilities achieved in 2-player tournaments, for different spitefulness values, s.

by more spiteful agents. However, in contrast, we see that, in this 2 player tournament, the utility of our opponent decreases more rapidly than for our agent. Therefore in this setting, a more spiteful approach is more desirable as we attempt to maximise our score relative to that of the opponents.

If we consider the scenarios with slight discounting ($\delta = 0.75$) and a small utility of conflict ($u_{\overline{\alpha}} = 0.25$), we see that altering the spitefulness parameter can affect whether or not our utility exceeds that of the opponent. In particular, Figure 4.3 shows that if $s < \frac{1}{4}$ our agent is slightly beaten by its opponent, but for $s > \frac{1}{4}$, our agent slightly beats its opponent.

These results show that, against a single opponent, it is effective to take a spiteful approach in order to beat that opponent. However, in a tournament with many opponents, it is detrimental to use such an approach. Therefore, in the remainder of this chapter (where we consider negotiation tournaments with many players), we use a non-spiteful version of our agent (with $s = 0$).

4.3 Results of the Automated Negotiating Agents Competitions

In this section, we briefly present the results of the 2010 (Section 4.3.1), 2011 (Section 4.3.2) and 2012 (Section 4.3.3) editions of the Automated Negotiating Agents Competition, which the agents we developed participated in.

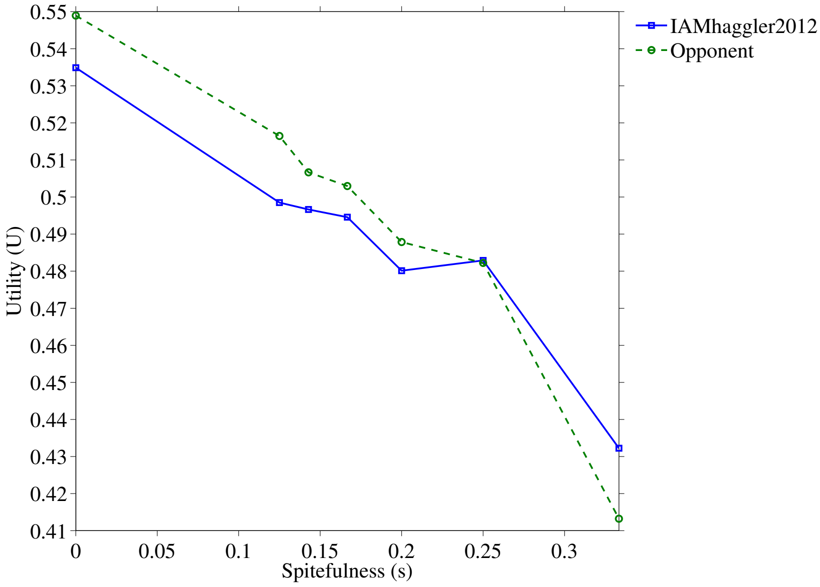


FIGURE 4.3: Average utilities achieved in 2-player tournaments (with $\delta = 0.75$ and $u_{\overline{\alpha}} = 0.25$), for different spitefulness values, s.

4.3.1 ANAC 2010

The first Automated Negotiating Agents Competition (ANAC 2010) was held at the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010, Toronto, Canada). The competition consisted of a tournament between 7 agents from 5 institutions (as listed in Table 4.2). The strategy described in Section 3.7.1 was entered under the name 'IAMhaggler', whilst the *IAMcrazyHaggler2010* strategy described in Section 3.7.4 was entered under the name 'IAMcrazyHaggler'.

During the competition, negotiation sessions were run using the GENIUS environment (see Section 2.2.1). These sessions were run between all two-party combinations of the 7 agents, excluding self-play. Each pair of agents negotiated using each preference profile in the scenario, resulting in a total of 42 sessions per scenario.

Agent Name(s)	Affiliation
Agent K Nozomi	Nagoya Institute of Technology, Japan
Yushu	University of Massachusetts Amherst, USA
IAMhaggler IAMcrazyHaggler	University of Southampton, UK
FSEGA	Babes Bolyai University, Romania
AgentSmith	Delft University of Technology, Netherlands

TABLE 4.2: Participants in the Automated Negotiating Agents Competition 2010. Source: Baarslag et al. (2010)

Rank	Agent	Itex - Cypress	England - Zimbabwe	Travel	Mean
1	**Agent K**	0.901	0.712	**0.685**	**0.766**
2	Yushu	0.662	**1.000**	0.250	0.637
3	Nozomi	**0.929**	0.351	0.516	0.599
4	IAMhaggler	0.668	0.551	0.500	0.573
5	FSEGA	0.722	0.406	0.000	0.376
6	IAMcrazyHaggler	0.097	0.397	0.431	0.308
7	Agent Smith	0.069	0.053	0.000	0.041

Table 4.3: Scores achieved in the Automated Negotiating Agents Competition 2010. Source: Baarslag et al. (2010)

Table 4.3 shows the scores achieved by each agent that participated in the competition. The winner was *Agent K*, developed at the Nagoya Institute of Technology, which achieved an average score of 0.766. Our agent, *IAMhaggler2010*, performed fairly consistently across the scenarios, finishing in third place in the *England vs Zimbabwe* and *Travel* scenarios, and in fourth place in the *Itex vs Cypress* scenario. Overall, it finished in 4th place. Despite the simplicity of our additional agent, *IAMcrazyHaggler2010*, it outperformed some of the other agents, particularly in the larger two scenarios, where it finished in 4th and 5th place.

4.3.2 ANAC 2011

A total of 18 agents, from 7 institutions were entered into the second Automated Negotiating Agents Competition (ANAC 2011), which was held at the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011, Taipei, Taiwan). Due to the large number of participants, the competition consisted of a qualifying round, and a final containing the top 8 agents from the qualifying round. Our agent, *IAMhaggler2011*, successfully reached the final and went on to finish in third place. Our additional agent, *IAMcrazyHaggler2011* did not qualify for the final, but finished in 16th place. The results of the final round are given in Table 4.4. It should be noted that there is little difference amongst the results of the agents which finished in 4th, 5th and 6th places.

4.3.3 ANAC 2012

A total of 17 agents, from 8 institutions were entered into the third Automated Negotiating Agents Competition (ANAC 2012), which was held at the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012, Valencia, Spain). In common with ANAC 2011, due to the large number of participants, the competition again consisted of a qualifying round, and a final containing the top 8 agents

Rank	Agent Name	Affiliation	Score
1	**HardHeaded**	**TU Delft, Netherlands**	**0.743**
2	Gahboninho	Bar-Ilan University, Israel	0.728
3	IAMhaggler2011	University of Southampton, UK	0.683
4	BRAMAgent	Ben-Gurion University of the Negev, Israel	0.675
5	AgentK	Nagoya Institute of Technology, Japan	0.672
6	TheNegotiator	Delft University of Technology, Netherlands	0.671
7	Nice Tit for Tat Agent	Delft University of Technology, Netherlands	0.665
8	ValueModelAgent	Bar-Ilan University, Israel	0.607

Table 4.4: Scores achieved in the final round of the Automated Negotiating Agents Competition 2011.

Rank	Agent Name	Affiliation	Score
1	**CUHKAgent**	**Chinese University of Hong Kong**	**0.626**$\pm$0.001
2	AgentLG	Bar-Ilan University, Israel	0.622$\pm$0.001
3-4	OMACagent	Maastricht University, Netherlands	0.618$\pm$0.001
3-4	TheNegotiator Reloaded	Delft University of Technology, Netherlands	0.617$\pm$0.001
5	BRAMAgent2	Ben-Gurion University of the Negev, Israel	0.593$\pm$0.001
6	Meta-Agent	Ben-Gurion University of the Negev, Israel	0.586$\pm$0.001
7	IAMhaggler2012	University of Southampton, UK	0.535$\pm$0.000
8	AgentMR	Nagoya Institute of Technology, Japan	0.328$\pm$0.001

Table 4.5: Scores achieved in the final round of the Automated Negotiating Agents Competition 2012, including 95% confidence intervals.

from the qualifying round. Our agent, *IAMhaggler2012*[2], successfully reached the final and went on to finish in seventh place. Our additional agent, *IAMcrazyHaggler2012* also qualified for the final (in a lower position than *IAMhaggler2012*), but we chose to withdraw it from the final. The results of the final round are given in Table 4.5. The statistical significance of the results was calculated using Welch's t-test (Welch, 1947) to test for the null hypothesis, given the mean, variance and number of results in our sample. Welch's t-test is an extension of Student's t-test (Student, 1908) for comparing samples in which the variance may differ, as in the results we consider. Using this test, it was found that the agents which finished in 3rd and 4th places had scores that were not statistically significantly different from each other. Therefore both agents were awarded a prize for finishing in joint third place. Differences between all other positions were found to be statistically significant.

Having discussed the results of recent international negotiating agent competitions, our evaluation now focuses on the results of our own experiments, firstly considering the performance of agents in self play.

[2]In the competition, a preliminary version of *IAMhaggler2012* was used, which did not set the reservation value depending on the scenario, as described in Section 3.6.

4.4 Self Play

Whilst our strategies are designed for negotiations against an unknown opponent, since that opponent may be using a similar, or even the same strategy as our agent, it is important that our strategy performs well even in self-play. Therefore, we begin our evaluation by considering the performance of our strategy in negotiations where the opponent uses the same strategy as our agent. In such a setting, it may be possible that a 'tough' strategy, which does not concede at all, would fail to reach agreement with a similar agent, even though it may perform well if enough of the other strategies concede far enough to reach agreement with it. As an example, *IAMcrazyHaggler2010* proposes offers which have a utility greater than 0.95, and accepts those with a utility greater than 0.90. In many of the domains we consider, there are no outcomes that give one agent a utility of above 0.95 while the other achieves more than 0.9. Consequently, in these domains, *IAMcrazyHaggler2010* will be unable to reach an agreement. By evaluating the agents in self-play, we show the co-operativeness of various strategies.

For each agent, we run a single negotiation for each scenario and the mean score of the two agents is then taken. This is repeated 3 times in order to increase the confidence of the results. Table 4.6 shows these results, for all agents, averaged across all 216 scenarios. The 95% confidence intervals are also given.

Of all of the agents tested in this part of the evaluation, it was our agent, *IAMhaggler2012* which achieved the highest score. Specifically, it reached a utility 87.0% of the maximum possible.[3] By analysing this further, we find that the raw utility (before discounting) of the outcomes reached by *IAMhaggler2012* was 94.4% of the maximum possible. Only *AgentLG* performed significantly better under this measure, achieving 95.6% of the maximum possible. Not only does it reach highly efficient agreements, it also does so in reasonable time. Specifically, the average time of agreement (or conflict) of our agent was 0.50 (i.e. half way to the negotiation deadline). Compared to the other agents, only the *IAMcrazyHaggler2012* agents reached earlier agreements on average, with *AgentLG*'s average time of agreement being 0.76. Overall, in self-play, both instances of *IAMhaggler2012* are trying to concede as a best response to their opponent, and this feedback loop results in relatively fast concession which leads to an early agreement which is also quite efficient.

In this evaluation, we included three variants of *IAMcrazyHaggler2012*, each with a different threshold value, $U_{\min} \in \{0.7, 0.8, 0.9\}$. The agent makes offers that lie above its threshold value and accepts those that lie no lower than 2% below the threshold

[3]The maximum possible self-play utility is found by taking the average utility of the two parties in the utilitarian solution to each scenario (i.e. the solution which maximises the sum of the utilities), averaged over all scenarios. In the scenarios we consider, this value is 0.79.

Agent	Self-play Score	% of maximum
Maximum	*0.793* ±*0.000*	*100.0* ±*0.0*
IAMhaggler2012	0.690 ±0.002	87.0 ±0.2
TheNegotiator Reloaded	0.656 ±0.010	82.7 ±1.2
Meta-Agent	0.649 ±0.012	81.8 ±1.5
$\text{IAMcrazyHaggler2012}_{0.7}$	0.645 ±0.013	81.3 ±1.6
AgentLG	0.618 ±0.002	77.8 ±0.2
CUHKAgent	0.603 ±0.009	76.0 ±1.1
OMACagent	0.575 ±0.006	72.4 ±0.7
BRAMAgent2	0.573 ±0.006	72.2 ±0.7
$\text{IAMcrazyHaggler2012}_{0.8}$	0.539 ±0.009	67.9 ±1.2
AgentMR	0.263 ±0.001	33.2 ±0.1
$\text{IAMcrazyHaggler2012}_{0.9}$	0.249 ±0.001	31.3 ±0.1

Table 4.6: Self-play scores, across all 216 scenarios, with 95% confidence intervals.

value. We found that its results varied considerably according to this parameter. With a higher threshold, the agent is guaranteed to reach a high valued agreement, if such an agreement is possible. In contrast, with a lower threshold, the agent is more likely to reach an agreement, but potentially with lower value.

In more detail, of the 24 base scenarios (as listed in Table 4.1), only 2 of them have possible outcomes which can be reached by a pair of *IAMcrazyHaggler2012* agents with $U_{\min} = 0.9$. A further 11 have possible outcomes which can be reached if both agents have $U_{\min} = 0.8$, while a further 6 have possible outcomes which can be reached if both agents have $U_{\min} = 0.7$. The remaining 5 scenarios do not have outcomes which can be reached by *IAMcrazyHaggler2012* agents with thresholds of at least $U_{\min} = 0.7$ This is reflected in the results of *IAMcrazyHaggler2012*, with the 0.9, 0.8 and 0.7 variants reaching agreement in 2, 13 and 19 scenarios, respectively.

All variants of *IAMcrazyHaggler2012* are able to reach agreements very quickly, on average, just 1.5% into the negotiation (within 2.7 seconds). A limitation of using such a fixed strategy as the one used by *IAMcrazyHaggler2012* is that its performance depends highly on the competitiveness of the scenario. In a more competitive scenario, where there is no outcome for which both agents can achieve a utility above the agent's fixed threshold, the self-play score is zero. In contrast, if the scenario is less competitive, the agreements that are made may not be very efficient, since much of the outcome space could be accessible above the thresholds of both agents.

One of the requirements of our work was to produce an agent that can perform well in an environment where the behaviour of the opponent is unknown (Requirement 3). So far, this evaluation has only considered negotiations in which the strategy used by the

Our Agent \ Opponent Agent	AgentLG	AgentMR	CUHKAgent	OMACagent	IAMhaggler2012	BRAMAgent2	Meta-Agent	TheNegotiator Reloaded
AgentLG	0.618	0.340	0.650	0.593	0.694	0.593	0.694	0.706
AgentMR	0.309	0.263	0.296	0.288	0.294	0.294	0.350	0.394
CUHKAgent	0.579	0.351	0.603	0.625	0.640	0.620	0.691	0.716
OMACagent	0.600	0.333	0.614	0.575	0.682	0.638	0.676	0.688
IAMhaggler2012	0.633	0.339	0.619	0.576	0.690	0.636	0.643	0.664
BRAMAgent2	0.609	0.345	0.606	0.527	0.646	0.573	0.640	0.657
Meta-Agent	0.601	0.341	0.530	0.539	0.685	0.614	0.649	0.662
TheNegotiator Reloaded	0.646	0.341	0.596	0.589	0.704	0.636	0.663	0.656

TABLE 4.7: Scores achieved in our experiments, averaged over all scenarios.

opponent is the same as the one used by our agent. Therefore, we now consider the results of negotiations in which the strategies of the two players differ.

4.5 Extended Evaluation against other ANAC 2012 Agents

Our aim in this section is to evaluate the performance of our strategy against unknown opponents in a range of scenarios. Specifically, the set of agents that we use in this evaluation consists of the seven agents submitted by other institutions to ANAC 2012 (as listed in Table 4.5), plus the latest version of our *IAMhaggler2012* agent (as described in Section 3.7.3). The agents have been independently developed and represent the state-of-the-art in practical strategies for automated negotiation in multi-issue scenarios. The scenarios that we use are also taken from ANAC 2012, as listed in Table 4.1. These scenarios vary considerably, in terms of size, competitiveness, discounting factor and utility of conflict, ensuring that our analysis covers a wide variety of scenarios.

To measure the performance of the various strategies, we first carried out negotiation sessions between all pairs of strategies, in each scenario, using the University of Southampton's 'Iridis' computing cluster. Each negotiation session ran on a single core of a 6-core, 2.2 GHz processor, with 2 GB of RAM being allocated to the negotiation session. Furthermore, in order to reduce the significance of any random behaviour, and in doing so, create statistically significant results, we repeated each negotiation 3 times. Therefore, in our experiments, each agent carried out a total of 24 negotiations per preference profile. To complete the experiment, with 8 agents and 216 scenarios, a total of $8 * 8 * 216 * 3 = 41,472$ negotiation sessions were carried out. We present the score for each pairing of agents, averaged across all scenarios, in Table 4.7.

Rank	Agent	Score
1	AgentLG	0.610 ±0.000
2-3	OMACagent	0.605 ±0.000
2-3	CUHKAgent	0.603 ±0.002
4	TheNegotiator Reloaded	0.597 ±0.002
5	IAMhaggler2012	0.587 ±0.001
6	BRAMAgent2	0.576 ±0.002
7	Meta-Agent	0.567 ±0.003
8	AgentMR	0.318 ±0.001

Table 4.8: Scores across all scenarios, with 95% confidence intervals.

We can use these results to analyse the performance between any pair of agents. We can also generate tournament results (using the same approach as using the ANAC competitions) with any subset of the agents, in any individual scenario or set of scenarios.

Specifically, in a tournament, all players negotiate with all other players. Therefore, the tournament score for player p is given by:

$$\frac{\sum_{q \in P, p \neq q} U(p, q)}{|P| - 1} \tag{4.1}$$

where $U(p, q)$ is the score that player p achieves when negotiating with player q, and $|P|$ is the number of players.

In the remainder of this section, we discuss the aggregated results of negotiation sessions in each of the scenarios we introduced at the start of this chapter. We consider the results from a range of different tournament settings to demonstrate the aggregated performance of each agent against a range of state-of-the-art opponents.

4.5.1 Average Results

Table 4.8 shows the average scores achieved by each of the agents, averaged across all 216 scenarios. This table shows that *AgentLG* achieved the highest score, with *IAMhaggler2012* finishing in 5th place. The scores of most of the agents are very similar, with the top 7 scores all being within 0.05 of each other. *IAMhaggler2012*'s score is only 0.023 less than that of the winner.

We now analyse the results in more detail by considering sub-tournaments which contain only a subset of the scenarios. Specifically, we use the classifications introduced in Section 4.1, which partition the scenarios according to their discounting factor, utility of conflict, competitiveness and domain size. We now analyse each of these classifications in turn in Sections 4.5.2 to 4.5.5.

Agent	Score (Rank)							
	mean		$\delta = 0.50$		$\delta = 0.75$		$\delta = 1.00$	
AgentLG	**0.610**	**(1)**	0.518	(3-4)	**0.605**	**(1)**	0.707	(2-3)
OMACagent	0.605	(2-3)	0.510	(5-6)	0.599	(2-3)	0.705	(2-3)
CUHKAgent	0.603	(2-3)	0.526	(2)	0.597	(2-3)	0.686	(4)
TheNegotiator Reloaded	0.597	(4)	0.513	(3-4)	0.550	(7)	**0.726**	**(1)**
IAMhaggler2012	0.587	(5)	**0.538**	**(1)**	0.569	(5)	0.655	(5-6)
BRAMAgent2	0.576	(6)	0.508	(5-6)	0.593	(4)	0.627	(7)
Meta-Agent	0.567	(7)	0.496	(7)	0.561	(6)	0.645	(5-6)
AgentMR	0.318	(8)	0.337	(8)	0.369	(8)	0.247	(8)

Table 4.9: Scores and ranks across all scenarios, grouped by discounting factor, with winning scores marked in bold.

4.5.2 Effect of Discounting Factor

In this section, we isolate the effect of the discounting factor on the performance of the agents by averaging over the results of the scenarios with the same discounting factor. As mentioned in Section 4.1, the scenarios use discount factors $\delta \in \{0.50, 0.75, 1.00\}$.

Table 4.9 shows the results for each value of δ. A decrease in δ increases the effect of the discounting and, therefore, the average score of each agent decreases as δ decreases (except in the case of *AgentMR*). For all values of $\delta \in \{0.50, 0.75, 1.00\}$, *AgentMR* finishes in 8th place, and therefore all other agents achieve a higher score. Furthermore, *AgentLG* and *CUHKAgent* always achieve a higher score than *Meta-Agent* and *BRAMAgent2*. Finally, *OMACagent* always achieves a higher score than *Meta-Agent*.

Our agent is the winner of the tournament with the most severe discounting factor ($\delta = 0.50$), achieving a score 2.3% higher than that of the second place agent (*CUHKAgent*). For larger (less severe) discounting factors, our agent is less successful, finishing in 5th or 6th place, with scores 6% and 10% below those of the winning agent. Three different agents (*AgentLG*, *TheNegotiator Reloaded* and *IAMhaggler2012*) each win the tournament for one of the discounting factor values used.

The reason for our strong performance in such scenarios is that, when discounting is severe, there are considerable benefits to reaching an early agreement (or in cases where the utility of conflict is high, early conflict). By considering the effect of discounting on the value of future offers, *IAMhaggler2012* generally concedes more quickly in highly discounted scenarios, thereby reaching early agreements with a relatively high discounted utility. On the other hand, in less severely discounted scenarios (including those with no discounting), due to its adaptiveness to the behaviour of the opponents, *IAMhaggler2012* takes an approach which is more concessive than necessary, therefore it is outperformed by a number of other strategies.

Agent	Score (Rank)							
	mean		$u_{\overline{\alpha}} = 0.00$		$u_{\overline{\alpha}} = 0.25$		$u_{\overline{\alpha}} = 0.50$	
AgentLG	**0.610**	**(1)**	**0.594**	**(1)**	**0.610**	**(1)**	**0.626**	**(1-3)**
OMACagent	0.605	(2-3)	0.585	(2-3)	0.602	(2-3)	**0.627**	**(1-3)**
CUHKAgent	0.603	(2-3)	0.585	(2-3)	0.601	(2-3)	**0.623**	**(1-4)**
TheNegotiator Reloaded	0.597	(4)	0.575	(4-5)	0.595	(4)	0.620	(3-4)
IAMhaggler2012	0.587	(5)	0.575	(4-5)	0.578	(5-6)	0.609	(5)
BRAMAgent2	0.576	(6)	0.560	(6)	0.575	(5-6)	0.593	(7)
Meta-Agent	0.567	(7)	0.540	(7)	0.563	(7)	0.599	(6)
AgentMR	0.318	(8)	0.192	(8)	0.317	(8)	0.444	(8)

TABLE 4.10: Scores and ranks across all scenarios, grouped by utility of conflict, with winning scores marked in bold.

4.5.3 Effect of Utility of Conflict

We now consider the effect of the utility of conflict on the agents, using the same technique as used for the discounting factor. Our scenarios have utility of conflict values $u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$ and the results for each of these values are shown in Table 4.10. Unsurprisingly, the scores achieved by all agents tend to increase as the utility of conflict increases.

For all values of $u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$, our agent finished between 4th and 6th place, with similar performance being observed regardless of the utility of conflict. The top agent overall (*AgentLG*) won the tournament regardless of the utility of conflict value, with the agents which finished in joint 2nd place overall (*CUHKAgent* and *OMACagent*) also achieving joint first place in the tournament with the highest utility of conflict ($u_{\overline{\alpha}} = 0.50$).

In Table 4.11, we consider the proportion of negotiations which ended with a specific agent sending an END message to terminate the negotiation before the deadline[4]. Only 5 of the 8 agents were observed to send END messages. Specifically these were *OMACagent*, *TheNegotiator Reloaded*, *IAMhaggler2012*, *BRAMAgent2* and *Meta-Agent*. As expected, for each agent, the number of END messages did not decrease as $u_{\overline{\alpha}}$ increased and no agents sent END messages (except as noted in footnote 4).

4.5.4 Effect of Competitiveness

Table 4.12 shows the results of the agents in scenarios from each competitiveness class. As can be seen, more competitive domains result in lower average scores. Furthermore, the top three agents overall (*AgentLG*, *CUHKAgent* and *OMACagent*) each win in at

[4]Note that here we only consider END messages which were sent at least 1 second before the deadline. Some agents were found to send END messages in the final second when the discounting factor and utility of conflict parameters made it irrational to do so.

Agent	Percentage of scenarios ending in conflict		
	$u_{\overline{\alpha}} = 0.00$	$u_{\overline{\alpha}} = 0.25$	$u_{\overline{\alpha}} = 0.50$
AgentLG	0	0	0
OMACagent	0	0	1.2%
CUHKAgent	0	0	0
TheNegotiator Reloaded	0	0	5.2%
IAMhaggler2012	0	6.3%	26.5%
BRAMAgent2	0	0	2.4%
Meta-Agent	0	0.5%	6.6%
AgentMR	0	0	0

TABLE 4.11: END messages sent across all scenarios, grouped by utility of conflict.

Agent	Score (Rank)							
	mean		$C(O) \leq 0.22$		$0.22 < C(O)$ $C(O) \leq 0.30$		$0.30 < C(O)$	
AgentLG	**0.610**	**(1)**	0.723	(3)	**0.631**	**(1-2)**	0.476	(2)
OMACagent	0.605	(2-3)	0.708	(4-5)	0.621	(2-4)	**0.484**	**(1)**
CUHKAgent	0.603	(2-3)	**0.735**	**(1)**	**0.626**	**(1-4)**	0.449	(4)
TheNegotiator Reloaded	0.597	(4)	0.732	(2)	0.625	(2-4)	0.432	(6)
IAMhaggler2012	0.587	(5)	0.704	(4-5)	0.599	(5)	0.458	(3)
BRAMAgent2	0.576	(6)	0.696	(6-7)	0.589	(6-7)	0.443	(5)
Meta-Agent	0.567	(7)	0.695	(6-7)	0.588	(6-7)	0.419	(7)
AgentMR	0.318	(8)	0.416	(8)	0.290	(8)	0.246	(8)

TABLE 4.12: Scores and ranks across all scenarios, grouped by competitiveness, with winning scores marked in bold.

least one competitiveness class. Our agent's best performance is in the highly competitive class, where it finishes in 3rd place. In highly competitive scenarios, it is essential for the participants to concede enough in order to reach an agreement. Due to the adaptiveness of *IAMhaggler2012* towards its opponent's behaviour, it is very good at ensuring that it concedes enough so as to reach an agreement. However, in less competitive scenarios it may concede too quickly, or too far than is necessary to reach an agreeement, since it fails to consider the effect that its behaviour can have on an adaptive opponent. Specifically, if the opponent is adaptive (as many of them are), *IAMhaggler2012* could benefit from taking a less concessive approach than it currently does. This benefit would be particularly noticeable in less competitive scenarios.

4.5.5 Effect of Domain Size

We now consider the performance of the agents depending on the size of the scenario's outcome space. Table 4.13 shows the results of the agents in scenarios from each size class. The top four agents overall (*AgentLG*, *CUHKAgent*, *OMACagent* and *TheNegotiator Reloaded*) each win in at least one size class. Our agent's best performance is in the scenarios with the smallest outcome spaces, where it finishes in joint 2nd place.

Agent	Score (Rank)							
	mean		$\lvert O\rvert \leq 200$		$200 < \lvert O\rvert$ $\lvert O\rvert \leq 3500$		$3500 < \lvert 0\rvert$	
AgentLG	**0.610**	**(1)**	0.534	(2-4)	**0.668**	**(1-3)**	**0.627**	**(1)**
OMACagent	0.605	(2-3)	**0.547**	**(1)**	0.659	(4)	0.608	(2-3)
CUHKAgent	0.603	(2-3)	0.538	(2-4)	**0.670**	**(1-3)**	0.602	(3-4)
TheNegotiator Reloaded	0.597	(4)	0.517	(6)	**0.669**	**(1-3)**	0.604	(2-4)
IAMhaggler2012	0.587	(5)	0.535	(2-4)	0.640	(5-6)	0.586	(5)
BRAMAgent2	0.576	(6)	0.526	(5)	0.632	(6-7)	0.570	(6)
Meta-Agent	0.567	(7)	0.512	(7)	0.634	(5-7)	0.556	(7)
AgentMR	0.318	(8)	0.326	(8)	0.338	(8)	0.289	(8)

TABLE 4.13: Scores and ranks across all scenarios, grouped by size, with winning scores marked in bold.

By considering the performance of the agents in sub-tournaments consisting of various classes of scenario, we observe that, due to the way in which it adapts to the discounting factor and the behaviour of the opponents, our agent is particularly well suited to scenarios which are significantly affected by time discounting, as well as those which are highly competitive. These represent the more challenging scenario classes, since, in time discounted domains, it is desirable to reach agreements without unnecessary delay, and furthermore, in highly competitive domains there is a careful balance to be had between not conceding enough, risking conflict and conceding too much, reaching an agreement with low utility.

4.5.6 Pareto Efficiency of Agreements

In order to evaluate the efficiency of the strategies (Requirement 2), we consider the Pareto efficiency of the agreements reached by the agents. This is determined by measuring the shortest Euclidean distance from the agreement point to the Pareto frontier. As described in Section 2.2.3, the Pareto frontier is a line connecting all Pareto efficient outcomes. Table 4.14 shows that the average distance to the Pareto frontier for all agents is quite similar, ranging from 0.015 to 0.022, with our agent's agreements being, on average, a Euclidean distance of 0.018 from the Pareto frontier. However, in terms of the agreements that were furthest from the Pareto frontier, this measure varies amongst the different agents. Specifically, all of the agents have least efficient agreements between 0.164 and 0.536 from the Pareto frontier, but in this respect, our agent was one of the better performers, with its least efficient agreement being just 0.214 from the Pareto frontier. Furthermore, in total, 14.1% of agreements were on the Pareto frontier and for negotiations containing our agent, this figure was slightly higher at 14.9%. This shows that, despite using a random approach to selecting offers at a given utility level (rather than by modelling the opponent's utility function) our agent is still able to reach agreements that are of similar efficiency to those of the other agents.

Agent	Mean distance	Worst distance	Percentage at frontier
AgentLG	0.016	0.330	14.8%
OMACagent	0.017	0.223	17.0%
CUHKAgent	0.018	0.449	15.9%
TheNegotiator Reloaded	0.019	0.329	11.2%
IAMhaggler2012	0.017	0.214	14.9%
BRAMAgent2	0.022	0.203	15.0%
Meta-Agent	0.019	0.536	13.1%
AgentMR	0.015	0.164	10.8%

Table 4.14: Pareto efficiency of agreements.

In summary, our agent performs well in a wide variety of scenarios. It reaches agreements which are generally close to Pareto efficient, and its average utility is 96% of that of the overall winner, *AgentLG*. In contrast, the agent with the worst overall utility, *AgentMR* only achieves an average utility which is 52% of that of the overall winner. In a number of subtournaments, where the scenarios exhibit specific characteristics, our agent outperforms many of its opponents. Specifically, this is the case when the scenario is highly discounted ($\delta = 0.5$), and, to a lesser extent, when the scenario is small $|O| < 200$ or is highly competitive ($C(O) > 0.30$).

4.6 Offer Rate

In order to measure the tractability of our strategy (Requirement 7), we also consider the number of offers made per second by each agent. This gives an indication of the amount of time taken to compute an offer, and can be easily be compared across different agents and domains. We calculate this value for negotiations where the opponent used the same strategy, thereby ensuring that this measure is not biased by the offer rate of the opponent.

In our experiments, we imposed a minimum amount of time between each offer. Specifically, we set this to 10ms, therefore limiting the number of offers per second to 100. The reason for this restriction is twofold. Firstly, it attempts to bring the conditions of our experiments more in line with those used in ANAC 2010 and ANAC 2011. Specifically, during these earlier competitions, the Genius platform was used in a standard mode which displays the negotiation trace of the agents as the negotiation progresses. The updating of this graphical element caused a delay in the exchange of offers, which, in a scenario with a deadline (as in all of the scenarios we consider) can have a significant effect on the outcome. In contrast, during our experiments, to enable the negotiations to be carried out on the computing cluster, we used a modified version of the Genius platform, which lacked the graphical element, and therefore lacked this delay. Without

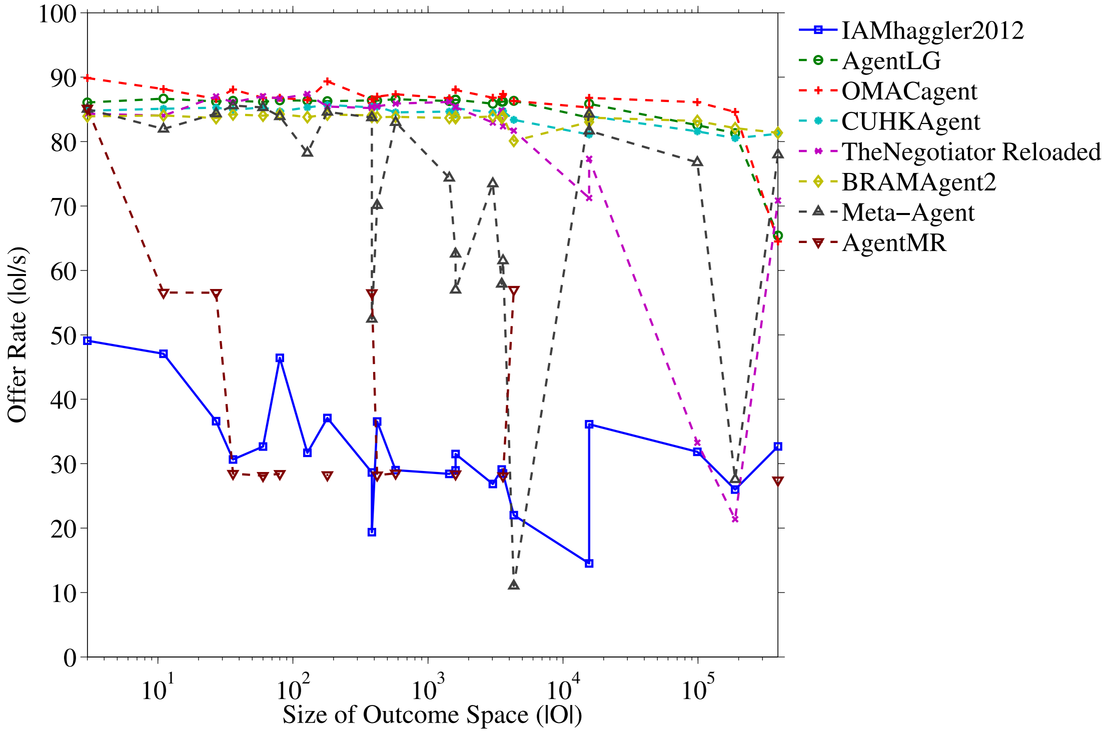


FIGURE 4.4: Offer rates (in offers per second) achieved by each agent, for scenarios with varying sizes of outcome spaces.

adding in the restriction, some of the agents are capable of making thousands of offers per second. Secondly, having such a delay is a natural model for many practical settings in which automated negotiation is likely to be used. For example, agents which negotiate over the internet will be restricted by the communication delay between them.

In more detail, Figure 4.4 shows the offer rate of each agent depending on the size of the scenario's outcome space. It shows that our agent has a considerably lower offer rate than that of the other agents. Specifically, the offer rate of our agent is between 14 and 49 offers per second, across all scenarios. In contrast, all of the other agents achieve offer rates in excess of 80 offers per second across many of the scenarios, and particularly those with smaller domains. Despite our agent being the slowest to produce offers, it is still able to produce many offers per second, even in the largest of the scenarios that we consider, thereby demonstrating that the approach it uses is computationally tractable.

4.7 Empirical Game Theoretic Analysis

A limitation of the tournament analysis that we performed as part of Section 4.5 is that it only considers the performance in a fixed set of tournaments. To demonstrate the stability of our strategies in a wide variety of tournaments, it is necessary to consider further tournaments in which the mix of strategies is different. We can then evaluate

	A	B	C
A	0.9	0.3	0.5
B	0.1	0.2	0.3
C	0.2	0.4	0.5

TABLE 4.15: Payoff matrix in an example game. The rows represent the different strategies an agent can take, whilst the columns represent the different strategies the opponent can take.

whether our strategy wins in such tournaments, and also identify whether any of the agents in those tournaments have an incentive to switch to a different strategy. To this end, we perform an empirical game theoretic analysis of the tournament results, using the technique introduced in Section 2.2.4. As already stated, empirical game theoretic analysis uses techniques from game theory in order to analyse empirical games, i.e. games in which the payoffs are determined by observations of the game (Wellman, 2006) rather than being part of the definition of the game. In this section, we first describe the methodology that we use in our empirical game theoretic analysis (Section 4.7.1), before presenting the results of that analysis (Section 4.7.2).

4.7.1 Methodology

In common with the approach developed by Jordan et al. (2007), we consider *single-agent deviations*, i.e. where there is an incentive for one agent to change its strategy, assuming that all other agents maintain their current strategy. We use this technique to search for Nash equilibria, which are defined as a combination of strategies such that there is no incentive for any of the players to change their strategy, given the strategies of the other players.

As an example, consider a tournament which consists of a two-player game between all pairs of players, as in the standard setup used in the Automated Negotiating Agent Competitions. Each player chooses the same strategy in all games in the tournament. Suppose that there is a choice of three known strategies, labelled A, B and C. Furthermore, suppose that the payoffs of a single game between two players is given by Table 4.15. In practice, each cell in this table is computed according to the average over the empirical outcomes of a number of negotiation sessions between a pair of agents which use the strategies corresponding to that cell.

Using the example, in a tournament of 5 players, if all players adopt strategy B, they will all achieve the score of 0.8 $(0.2*4)$. In contrast, if four of the players choose strategy B, but one of them chooses strategy C, the one choosing C will achieve a score of 1.6 $(0.4 * 4)$, whilst those choosing B will achieve a score of 0.9 $(0.2 * 3 + 0.3 * 1)$. This shows that there is an incentive, in a tournament where all players use strategy B, for one of the players to switch to C. By repeating this process it is possible to identify all single-player deviations, and consequently, Nash equilibria.

We present the deviations in our example game in Figure 4.5. Each node represents a possible mixture of the three strategies in a tournament. The vertices of the triangle represent strategy mixtures in which all agents play the same strategy. Furthermore, each arrow indicates that if a single agent deviates from the source mixture to the target mixture, then the additional payoff to that agent deviating will be positive, and statistically significantly different from zero. From any source mixture we only show the deviation (or deviations) which offer the highest such additional payoff. Therefore, we refer to these deviations as *statistically significant single-agent best deviations*. To measure the statistical significance of a deviation, we use Welch's t-test. Furthermore, the shaded strategy at each node represents the strategy which achieves the highest score and the Nash equilibria are the nodes which have no outgoing arrow. Our example in Figure 4.5 shows two Nash equilibria, in which all agents use strategy A or in which all agents use strategy C.

As well as looking at the deviations between different combinations of strategies, we can also consider the robustness of a Nash equilibrium by measuring the size of the *basin of attraction*. By this, we mean the number of strategy combinations that have a path of deviations leading to a specific Nash equilibrium. When doing so, it is necessary to consider that the strategy mixtures in Figure 4.5 are unevenly represented. Specifically, if all of the players were to choose a strategy at random (with equal probability), not all of the combinations would occur with equal probability. For example, there is only one way in which all six players could choose A, which we denote $AAAAAA$. In contrast, there are six ways that one of them can choose B, with the remaining four choosing A, which we could denote $BAAAAA, ABAAAA, ..., AAAAAB$. When calculating the size of the basin of attraction, we take this unevenness into account. In general, the relative measure of a node's contribution to this size is given by $\frac{n!}{a!b!c!}$ where n is the number of agents in the tournament and a, b and c are the number of agents which use strategies A, B and C, respectively.

In the example given in Figure 4.5, 21 of the nodes have their only path of deviations leading to the Nash equilibria at $AAAAA$. These 21 nodes represent 665 of the 729, or 91% of the possible combinations. Therefore, the $AAAAA$ equilibria has a basin of attraction which is 91% of the total space. A further Nash equilibria exists, at $CCCCC$, however this only represents 1 (or 0.13%) of the possible combinations. The remaining 63 (or 8.6%) of the possible combinations contain paths of deviation leading to both Nash equilibria. Since the $AAAAA$ equilibria has the larger basin of attraction, it is

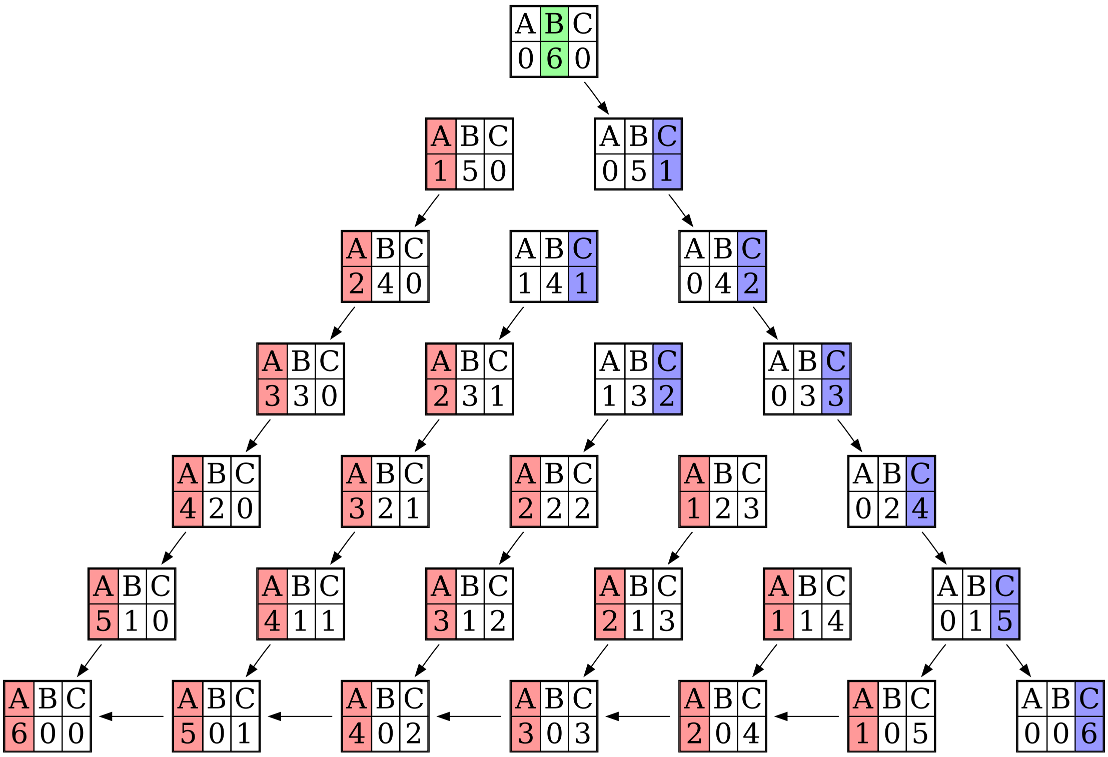


Figure 4.5: Deviation in example game, according to the payoff matrix in Table 4.15. Each node represents a combination of strategies in use in a tournament. Each node is a table, which displays, on the second row, the number of agents which use the strategy indicated on the first row. Each arrow is a single-player deviation which improves the score of the deviating player. The shaded strategies at each node indicate the strategy that achieves the highest score in that tournament.

considered to be the stronger of the two equilibria, indicating that strategy A is a strong choice. Furthermore, from Figure 4.5, we can also observe that if any player chooses strategy A, all agents should choose this strategy.

4.7.2 Results

To highlight some interesting effects which occur for certain sub-tournaments, we now perform a similar deviation analysis for the top three strategies, according to their average score across all scenarios. Therefore, we consider *AgentLG* (L), *OMACagent* (O) and *CUHKAgent* (C). Figure 4.6 shows the deviation analysis for this set of strategies. It shows that, from every 8-player tournament in which each agent selects one of the top three strategies, there exists a path of statistically significant deviations which lead to a single Nash equilibrium. This equilibrium is one in which all agents use the *AgentLG* strategy.

We also perform a similar analysis for each of the sub-tournaments of scenarios classified by discounting factor, utility of conflict, competitiveness and size, again considering the

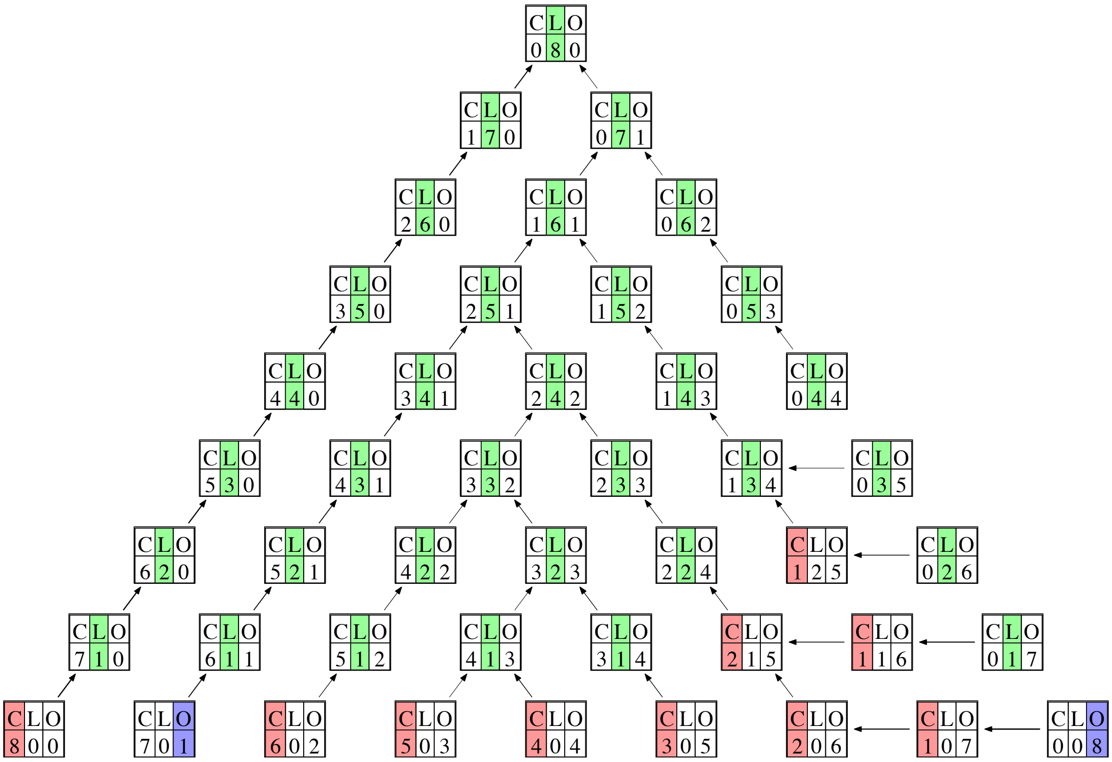


FIGURE 4.6: Deviation in the overall tournament (over all scenarios) for the strategies of *AgentLG* (denoted L), *OMACagent* (denoted O) and *CUHKAgent* (denoted C). The shaded strategies are the ones which achieve the highest scores.

top three agents in the respective sub-tournament. Our discussion focuses on the sub-tournaments which show more interesting results under this analysis. Specifically, we consider the sub-tournaments which contain scenarios with the following characteristics:

1. high discounting ($\delta = 0.50$) (*IAMhaggler2012*, *CUHKAgent*, *AgentLG*/*TheNegotiator Reloaded*)

2. low discounting ($\delta = 0.75$) (*AgentLG*, *OMACagent*, *CUHKAgent*)

3. no discounting ($\delta = 1.00$) (*TheNegotiator Reloaded*, *AgentLG*, *OMACagent*)

4. highly uncompetitive ($C(O) \leq 0.22$)[5] (*CUHKAgent*, *TheNegotiator Reloaded*, *AgentLG*)

5. highly competitive ($0.3 < C(O)$) (*OMACagent*, *AgentLG*, *IAMhaggler2012*)

Since the results in these sub-tournaments are generated as averages over a smaller set of scenarios, the values have a greater variance. Consequently, it is likely that a number of deviations will not provide a statistically significant change in utility for the deviating player. To ensure that our analysis continues to identify equilibria in these

[5] $C(O)$ is the competitiveness of the scenario, as defined in Section 4.1

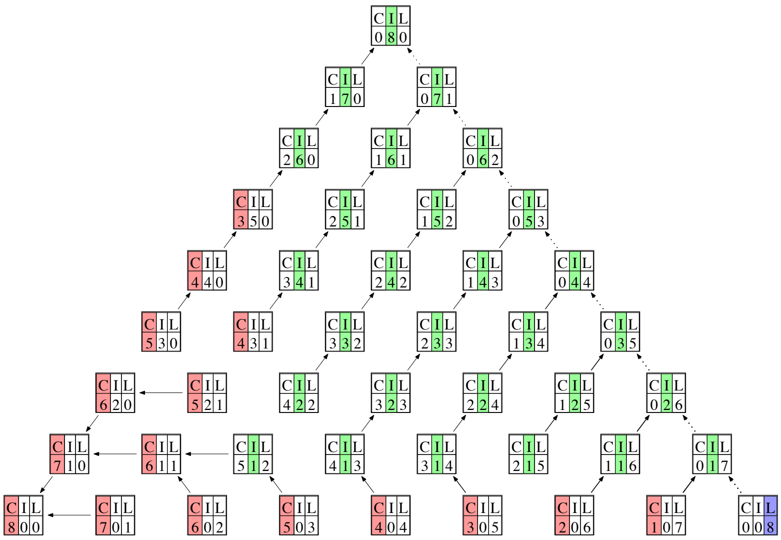
circumstances, we include these deviations, but identify them in our figures using a dashed line rather than a solid one.

In scenarios with high discounting ($\delta = 0.50$), there are two agents (*AgentLG* and *TheNegotiator Reloaded*) which achieve joint 3rd place. Therefore, for each of these strategies, we perform a separate analysis (Figures 4.7(a) and 4.7(b)). In each case we find two Nash equilibria, in which all agents use the *IAMhaggler2012* strategy, or in which all agents use the *CUHKAgent* strategy. Furthermore, if we perform the analysis with the top four strategies, the set of Nash equilibria is the same. Since we have multiple equilibria, we can consider the size of the basin of attraction of each equilibria. With either *AgentLG* or *TheNegotiator Reloaded* present, *IAMhaggler* has the largest basin of attraction. When *AgentLG* is present, the *IAMhaggler2012* equilibrium's basin of attraction represents 92% of the possible tournaments. When *TheNegotiator Reloaded* is present instead, this figure is reduced to 53%. In either case, from any tournament in which at least 3 of the 8 agents use *IAMhaggler2012*'s strategy, the only path of best single-agent deviations leads to the Nash equilibrium containing only that strategy.

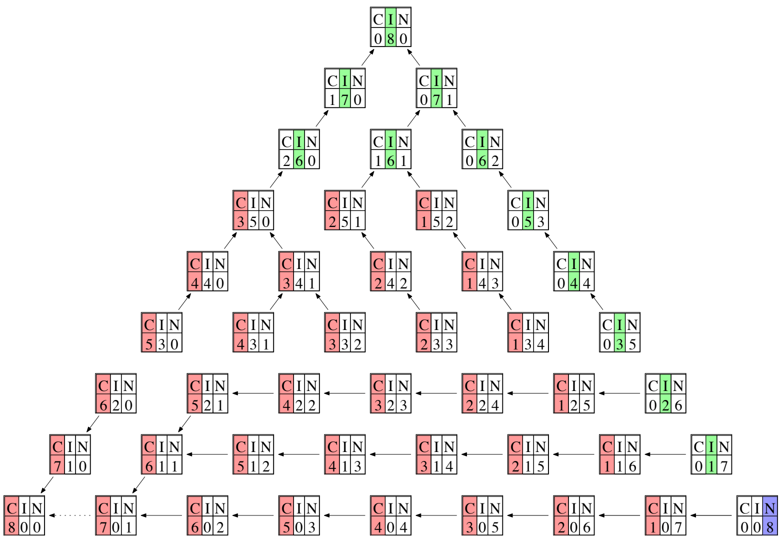
With low discounting ($\delta = 0.75$), we find a single Nash equilibrium (Figure 4.7(c)) in which all agents use the *AgentLG* strategy. With no discounting ($\delta = 1.00$), we again find a single Nash equilibrium (Figure 4.7(d)), in which half of the agents use *TheNegotiator Reloaded*'s strategy and the other half use *AgentLG*'s strategy.

In the highly uncompetitive scenarios ($C(O) \leq 0.22$), we find a single Nash equilibrium (Figure 4.7(e)) in which all three strategies are present, with 4 agents using *TheNegotiator Reloaded*'s strategy, 3 using *AgentLG*'s and 1 using *CUHKAgent*'s. In the highly competitive scenarios ($0.3 < C(O)$), we find two Nash equilibria (Figure 4.7(f)), each with a mixture of strategies. The one with a basin of attraction of 80% of the tournaments consists of 6 agents using the *IAMhaggler2012* strategy, while the remaining 2 use *AgentLG*'s. The other equilibrium consists of 5 agents using *AgentLG*'s strategy, whilst the remaining 3 use *OMACagent*'s.

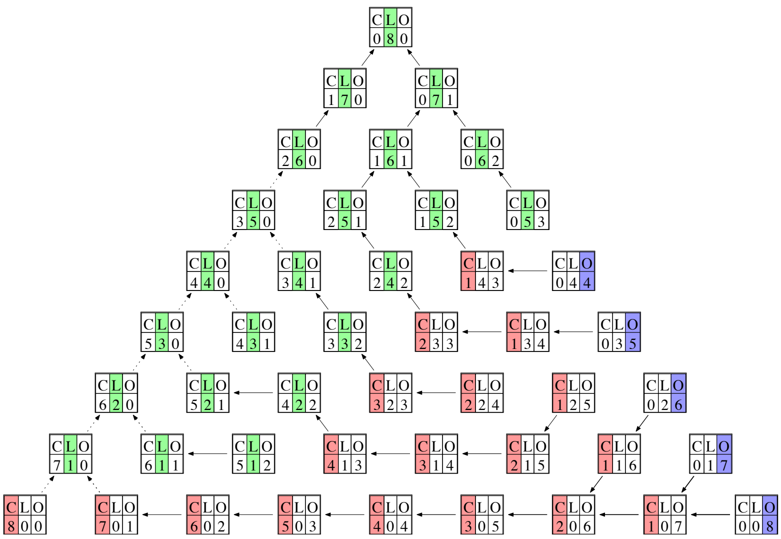
Overall, our empirical game theoretic analysis shows that, whilst in some sub-tournaments, there exists an equilibrium in which all agents use the strategy with the highest tournament score, there are others where such an equilibrium does not exist. Instead, in these sub-tournaments, we find equilibria consisting of multiple different strategies. The analysis also highlights the strength of *IAMhaggler2012* in highly discounted scenarios in that the equilibrium containing only our strategy also have a large basin of attraction. Specifically, provided at least 3 out of the 8 participants in the tournament use our strategy, the rest of them also have an incentive to use that strategy.



(a) $\delta = 0.50$, *IAMhaggler2012* (I), *CUHKAgent* (C) and *AgentLG* (L)

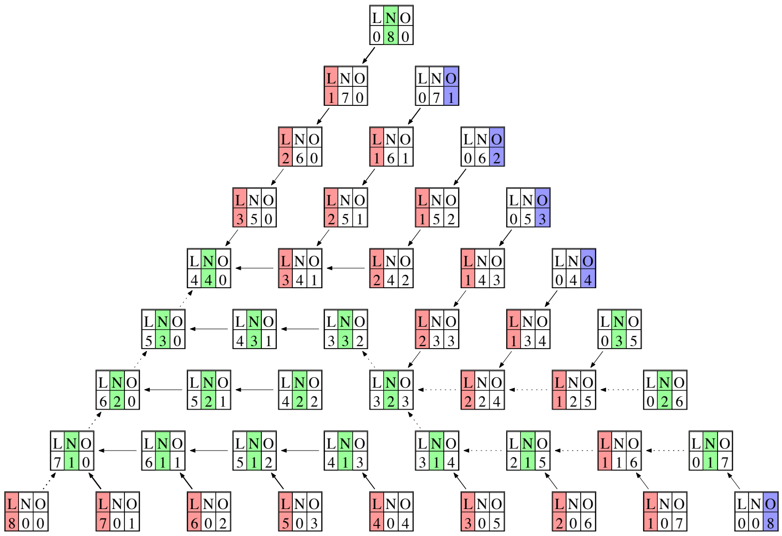


(b) $\delta = 0.50$, *IAMhaggler2012* (I), *CUHKAgent* (C) and *TheNegotiator Reloaded* (N)

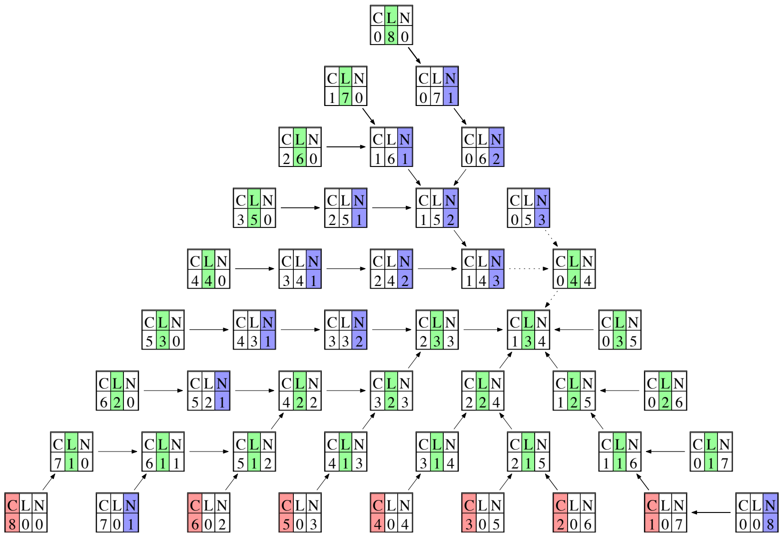


(c) $\delta = 0.75$, *AgentLG* (L), *OMACagent* (O) and *CUHKAgent* (C)

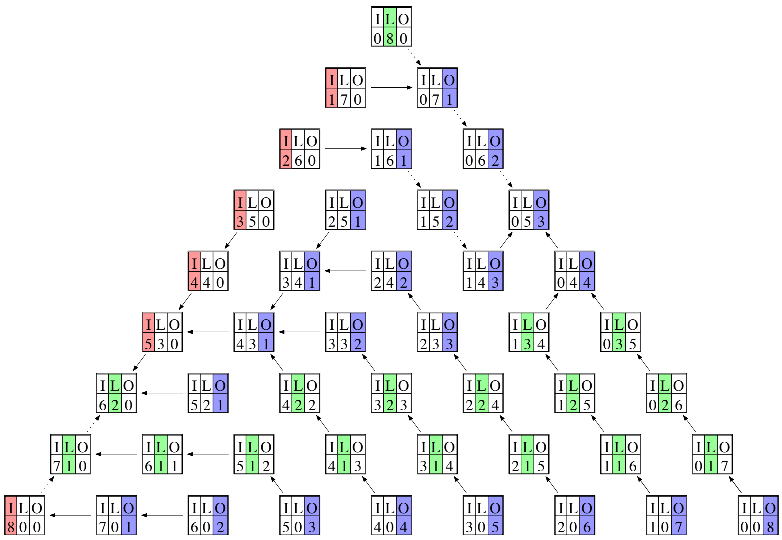
FIGURE 4.7: Deviation in a range of tournaments (with different subsets of scenarios) for the top three strategies in the respective tournament. The shaded strategies are the ones which achieve the highest scores.



(d) $\delta = 1.00$, *TheNegotiator Reloaded* (N), *AgentLG* (L) and *OMACagent* (O)



(e) $C(O) \leq 0.22$, *CUHKAgent* (C), *TheNegotiator Reloaded* (N) and *AgentLG* (L)



(f) $0.3 < C(O)$, *OMACAgent* (O), *AgentLG* (L) and *IAMhaggler2012* (I)

FIGURE 4.7: Deviation in a range of tournaments (with different subsets of scenarios) for the top three strategies in the respective tournament. The shaded strategies are the ones which achieve the highest scores. (continued)

4.8 Summary

In this chapter we have shown the performance of our agent against a number of other state-of-the-art agents across a wide range of scenarios. Specifically, in Section 4.1, we introduced the scenarios that are used throughout our evaluation. Then, in Section 4.2 we determined how to set our spitefulness parameter, showing the benefits of being spiteful in negotiation tournaments with only 2 players, but also showing how spiteful behaviour is detrimental in larger tournaments. Next, in Section 4.3 we briefly presented the results of the Automated Negotiating Agents Competitions, showing that *IAMhaggler2012* finished in fifth place in the most recent competition. Then, in Section 4.4, by considering the behaviour of the strategies in self-play, we have demonstrated the extreme sensitivity of our fixed strategy (*IAMcrazyHaggler*) to its acceptance threshold. Furthermore, we have shown that our adaptive strategy, *IAMhaggler2012*, outperforms the other strategies in self-play in many of the domains. In Section 4.5, we considered the results in negotiations against a range of state-of-the-art agents from the earlier competition, in order to demonstrate the performance of our agents against unknown opponents (Requirement 3). In terms of the average utility levels achieved by the agents, our agent generally performed well by adopting a concession strategy that adapts to the behaviour of the opponent. In particular, in scenarios where discounting had a significant effect, its average utility in a tournament setting was shown to be 2.3% higher than the agent with the second highest average utility. Furthermore, we have shown that all of the agents, including our own, tend to reach agreements that are Pareto efficient, and therefore meet Requirement 2. We also show that *IAMcrazyHaggler* and *IAMhaggler2012* fully meet our requirement for reaching agreements in domains where the ordering of each issue is unknown (Requirement 5). In Section 4.6, we showed that, in all of the scenarios that we tested, our agent was able to make a reasonable number of offers, and its offer rate was not significantly reduced in the scenarios with larger outcome spaces, showing that its approach is computationally tractable and therefore meets Requirement 7. By presenting an empirical game theoretic analysis of the top strategies from a number of different sub-tournaments, in Section 4.7 we have shown how, in a tournament setting consisting of scenarios where discounting has a significant effect, where all agents can choose one of the top three strategies in that tournament, there is a strong incentive for all agents to use our strategy. This result suggests that, for certain types of scenario, (in particular those with significant discounting) even when the negotiation environment contains other high performing agents, it is still appropriate to use the one we have developed.

Chapter 5

Design of Many-to-Many Negotiation Agents

In the previous two chapters, we have presented the design (Chapter 3) and evaluation (Chapter 4) of our one-to-one negotiation agents. In this chapter, we extend that work to consider negotiation environments which contain more than two parties. Such environments contain additional challenges over those present in the two-party ones. Specifically, each agent needs to carefully coordinate their behaviour against each opponent (Requirement 8). The opponents may vary, in terms of either their preferences or their behaviour. Furthermore, in some concurrent negotiation settings, it may be possible for one party to decommit from an existing agreement, subject to payment of a penalty (Requirement 9).

In the remainder of this chapter, we introduce the problem of concurrent negotiation that we consider in this work (Section 5.1). We then discuss the approach that we have developed for coordinating concurrent negotiation with a range of opponents, which we divide into two major components (Sections 5.2 and 5.3). Then, we discuss how to handle decommitment (Section 5.4). Finally, we describe the concurrent negotiation agent that we have formed from these strategies (Section 5.5). We summarise the chapter in Section 5.6.

5.1 Overview

Many-to-many negotiation, in which a set of parties negotiate with each other is more complex than a one-to-one negotiation. Specifically, a more advanced negotiation protocol is required in order to control the actions of the various parties. The many-to-many

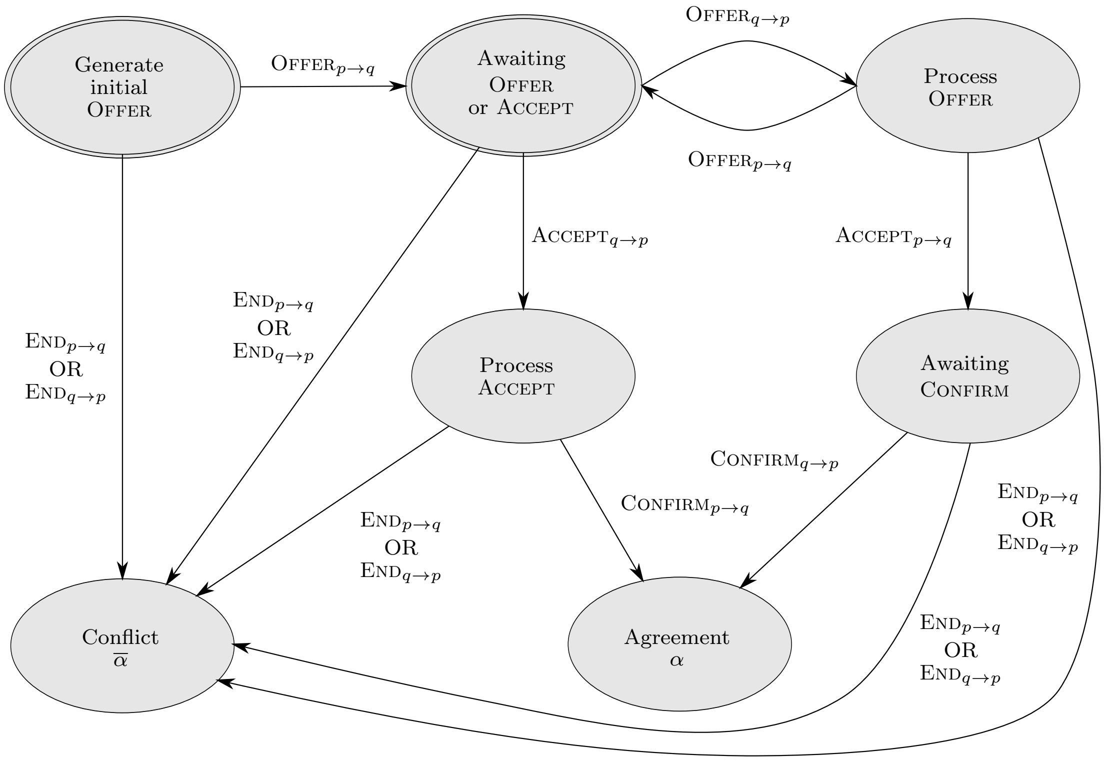


FIGURE 5.1: State diagram showing the concurrent negotiation protocol, from the perspective of a single negotiation thread of agent p.

negotiation protocol we consider in this work is similar to the ones described in An et al. (2009) and in Nguyen and Jennings (2005). Furthermore, as in Nguyen and Jennings (2005) we allow for *decommitment*, subject to a penalty, to allow for more flexibility and a fair comparison with the benchmark strategy.

In more detail, the negotiation considered in this chapter takes place in multiple, concurrent threads, between pairs of agents. In each of these threads, the agents use an alternating offers protocol, similar to the one used in the one-to-one negotiations in Chapter 3. As before, each offer represents a complete package, specifying the values for all negotiable issues, but with in our concurrent protocol, additional actions are possible. Specifically, the possible actions are OFFER, ACCEPT, CONFIRM, END and DECOMMIT. The negotiation begins with the agents exchanging OFFER messages. Sending an OFFER message in response to an OFFER from the opponent constitutes a counteroffer and implicitly a rejection of the previous offer. If an agent is satisfied with the most recent OFFER it received, it can send an ACCEPT message in order to indicate that it wishes to form an agreement. Following an ACCEPT message being sent in a negotiation thread, no further OFFER messages can be sent. Figure 5.1 is a state diagram from the perspective of a single negotiation thread, showing the messages exchanged and the various states of the thread.

In the standard alternating offers protocol, used in Chapter 3, the Accept message resulted in the formation of an agreement, and marked the end of the negotiation. In contrast, under the protocol used in this chapter, the Accept message does not necessarily result in an agreement. Instead, the negotiation moves into a new phase, in which the only messages allowed are Confirm and End. The Confirm message is used to indicate that the agent confirms that a binding agreement has been formed, whereas the End message will abort the negotiation thread.

The reason for including a Confirm message is as follows. Under this protocol, an agent is allowed to send offers to multiple opponents at once. Therefore, it may find that, while waiting for a response from them, more than one of these offers are accepted. If the Accept messages were to form a binding agreement at this point, the agent may inadvertently reach more than one agreement, and it would need to decommit from all but one of them, thereby incurring decommitment penalties. In this case, the Confirm and End messages can be used to select only one of them. Note that an agent could use this strategically by delaying sending the Confirm message. However, the agent is expected to confirm the acceptance within a short period of time (at most few seconds, depending on communication delays), otherwise it becomes invalid. Moreover, the opponent who sent the acceptance is still free to abort the agreement without penalty by using the End message. Provided that an agent does not Accept an opponent's offer whilst it is waiting for another agent to Confirm an acceptance (or End a negotiation), the agent can avoid reaching multiple agreements.

In a negotiation where there are multiple opponents, it is possible that, after a binding agreement is reached, one of the remaining opponents makes (or accepts) an offer that has a greater utility than that of the existing agreement. In such a situation, it may be beneficial to accept the new offer, and, at the same time, Decommit from the existing agreement. In order to discourage the agents from decommiting unnecessarily, we introduce a decommitment penalty, which is paid by the agent that chooses to decommit from a binding agreement. Without such a penalty, all agreements would essentially become non-binding, leading to a potentially unstable system. Note that, before a Confirm message has been sent within a thread, the agreement is not yet binding and so it is possible for either agent to send an End message to retract the offer without penalty.

In common with the work presented in the previous chapters, in our many-to-many negotiation scenarios, we also use a deadline and a discounting factor, which are both common to and known by all agents. As before, they are both measured in real time.

Figure 5.2 shows an example negotiation trace with three agents, where agent a negotiates concurrently with agents b and c. After a sends an offer to b, agent b accepts a's offer. Agent a then confirms and an agreement is reached. Agents a and c continue to

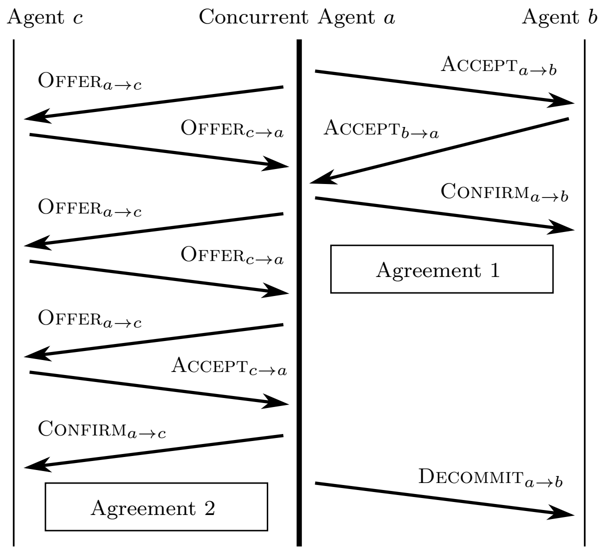


FIGURE 5.2: Sequence diagram showing a simplified negotiation trace between three agents, including two agreements and a decommitment.

negotiate, aiming to find an agreement that is better than the existing one (taking into account the decommitment penalty). After a total of five offers have been exchanged, agent c accepts a's offer. Agent a then confirms this agreement, and simultaneously decommits from the worse agreement with agent b. In practice, negotiation traces are likely to be considerably longer.

Having defined a protocol for concurrent negotiation, we now describe the strategy that we have developed for negotiating under such a protocol. Our strategy can broadly be split into two major components: the *negotiation thread managers* which handle the negotiation with a single opponent (Section 5.2) and the *coordinator*, which coordinates the behaviour of the set of negotiation thread managers (Section 5.3), which we now discuss in turn.

5.2 The Negotiation Thread Managers

The strategy of each negotiation thread manager is an extension of the one-to-one negotiation strategy detailed in Section 3.2.2. In more detail, each thread manager performs Gaussian process regression in order to predict the future concession of its opponent, i. The prediction is based on the offers received so far by this opponent, and is updated as more offers are received. The Gaussian process enables the prediction to be captured in the form of a probability distribution over the utility, $p_(u; \mu_{i,t}, \sigma_{i,t})$, for all future time points, $t \in [t_c, 1]$ (here, as before, time is normalised such that $t = 0$ represents the start of the negotiation and $t = 1$ represents the deadline). These probability distributions are then passed on to the coordinator, which uses them (along with those from other thread managers) to determine, for each opponent i, the best time, t_i^*, at which to reach an agreement, and the best utility, u_i^*, at which the thread manager should aim to reach the agreement. The way in which the coordinator calculates these values forms a core part of the negotiation strategy, and is discussed in detail in Section 5.3. For now, we will simply take these two values as given.

Given its target time, t_i^*, and target utility, u_i^*, a negotiation thread manager needs to: i) determine the target utility at which to generate offers and to accept incoming offers *right now* and, ii) generate multi-issue offers at the current target utility. Now, to determine the target utility, u_τ, at the current time, t_c, each thread manager uses polynomial time-dependent concession, where the concession rate is set such that the target utility level reaches u_i^* at time t_i^*. The time-dependent concession function taken by our many-to-many negotiation strategy mirrors the one used by our one-to-one negotiation strategies (described fully in Section 3.2.1). In summary, the target utility is given by:

$$u_\tau = U_0 - (U_{\min} - U_0) \cdot \begin{cases} t_c & \text{if } 0 \le t_c < 0.1 \\ \left(\dfrac{t_c(t_c - 0.1) + t_c^{1/\beta'}(0.2 - t_c)}{0.1} \right) & \text{if } 0.1 \le t_c < 0.2 \\ t_c^{1/\beta'} & \text{otherwise} \end{cases} \tag{5.1}$$

where U_0 is the utility of the initial offer, $U_{\min}$ is the agent's reservation utility (determined depending on the scenario using the approach described in Section 3.6) and t_c is the current time. β' is the constrained value of β, given by:

$$\beta' = \max(\beta_{\min}, \min(\beta_{\max}, \beta)) \tag{5.2}$$

where $\beta_{\min}$ and $\beta_{\max}$ are respectively the minimum and maximum values for β'. In our agent, we set $\beta_{\min} = 0.01$ and $\beta_{\max} = 2.0$. The reasons for constraining β in this way, and for the choice of $\beta_{\min}$ and $\beta_{\max}$ are the same as in Section 3.2.1. That is, the choice of the values themselves are somewhat arbitrary, but the aim is to avoid extreme behaviour whilst allowing flexibility. Finally, β is given by:

$$\beta = \frac{\log(t_i^*)}{\log\left(\dfrac{1 - u_i^*}{1 - U_{\min}} \right)} \tag{5.3}$$

where u_i^* and t_i^* are the target time and utility provided to the negotiation thread manager.

Finally, since we are concerned with multi-issue negotiation, it is necessary to generate a multi-issue offer, o, such that $U(o) \approx u_\tau$. To do so, we use the approach described in Section 3.3.1 which is to generate random offers until one is found which has a utility,

$U(o) \in [u_\tau - 0.025, u_\tau + 0.025]$. If an offer cannot be found within this range, the range is expanded, until a solution is found. Furthermore, if the target drops below the highest value of the offers made by the opponent, we instead propose the package with that utility that was offered by the opponent, as described in Section 3.3.2. This is since we assume that, for a set of possible offers with utility greater than u_τ, the one which is most likely to be accepted is the one which has previously been offered by the opponent. It may be possible to improve the selection of offers by modelling the preferences of the opponents (specifically, their utility functions over the multiple negotiation issues). However, due to the real-time aspect to the negotiations we consider, we found that using this simple, fast approach to selecting an offer produced very good results.

5.3 The Coordinator

The role of the coordinator is to calculate the best time, t_i^*, and utility value, u_i^*, at that time, for each thread manager. To do so, it uses the probability distributions received from the individual thread managers, which predict future utilities offered by the opponents. In the following, we use $P(u; \mu_{i,t}, \sigma_{i,t})$ to denote the cumulative probability distribution function, which is the (predicted) probability that the utility of an offer by opponent i will be at least u at time t, and $p(u; \mu_{i,t}, \sigma_{i,t})$ is the corresponding density function. In addition, recall that the negotiations are many-to-many, and so the opponents may exit the negotiations prematurely if they reach an agreement elsewhere. Since these values cannot be learned during a single negotiation (but can be learned by experimentation from repeated negotiations), we assume that the coordinator has prior knowledge of $P_{\mathrm{c},i}(t, t_{\mathrm{c}})$, which denotes the probability that opponent i will still be in the negotiation at time $t > t_{\mathrm{c}}$, given that it is in the negotiation at the current time, t_{c} .

Our approach is based on the one described in Section 3.2.2, but here it has been extended for negotiations with more than one opponent. In more detail, to find the optimal strategy, we begin by computing the best time to reach agreement, and then consider the best utility (or utilities), to offer at that time. We do the first part by computing the *expected* utility of an agreement at a given time, and choose the time with the highest expected utility. Although the protocol allows for decommitment, when we compute the expected utility, we simplify the equations by implicitly assuming that we terminate all other threads once an agreement is reached.[1] As a result, a single

[1]In practice, we do continue to negotiate (as explained in Section 5.4) but this is not captured by the expected utility. In principle, the equations can be extended to include the additional expected utility from decommitment, but this can become computationally intensive to compute, and we leave this for future work.

best time, $t^* \in [t_c, 1]$, is computed for all negotiation threads, as follows:

$$t^* = \text{argmax}_{t \in [t_c,1]} EU_{\text{rec}}(t) \tag{5.4}$$

where $EU_{\text{rec}}(t)$ is the expected utility when reaching an agreement at time t, given by:

$$EU_{\text{rec}}(t) = \frac{1}{|A|} \sum_{i \in A} P_{c,i}(t, t_c) \int_0^1 p_{[0,1]}(u; \mu_{i,t}, \sigma_{i,t}) D(u,t) \mathrm{d}u \tag{5.5}$$

where A is the set of *remaining* negotiation threads (i.e. those that have not terminated), and $P_{c,i}(t, t_c)$ is as defined above. Note that the expected utility is computed as the *average* expected utility for each thread. This is because, since we implicitly assume no decommitment, the expected utility assumes we are committed to the first thread that gives us an agreement. Thus, if multiple opponents were to reach agreements at roughly the same time, there is an equal probability that any one of those agreements will be reached.

Given the target agreement time, t^*, we would like to find the optimal utility level for each thread at which to produce offers. By varying the level in each thread, it is possible for an agent to concede more towards a specific opponent with which an agreement seems likely, whilst taking a less concessive approach against other opponents in the hope that an agreement with a higher utility may be achieved. To this end, we first specify the expected utility for a given vector of utility levels, one for each (remaining) opponent. We calculate this by assuming that the probability distributions from the various threads are *independently* sampled[2]. Furthermore, as before, we implicitly assume that no decommitment is allowed.

Given this, the expected utility of proposing offers at utility levels $\vec{u}$ at time t can be expressed as:

$$EU_{\text{offer}}(\vec{u}, t) = \sum_{A' \in \mathcal{P}(A)} \left(f(\vec{u}, A') \prod_{i \in A'} P(u_i; \mu_{i,t}, \sigma_{i,t}) \prod_{i \in A \setminus A'} (1 - P(u_i; \mu_{i,t}, \sigma_{i,t})) \right) \tag{5.6}$$

where A is the set of remaining opponents, u_i is the utility of the offer made to opponent i, $\mathcal{P}(A)$ is the powerset of A, $P(u_i; \mu_{i,t}, \sigma_{i,t})$ is the probability that opponent i will accept an offer of utility u_i at time t. Note that the right part of the equation denotes the probability of reaching an agreement with *exactly* the agents in the set A' at time t. Then, $f(\vec{u}, A')$ is the utility obtained if this occurs. For the same reasons as given above, since we implicitly assume no decommitment, the utility of this event is given

[2] Note that this is a simplifying assumption and applies to settings where the opponents have widely different strategies and/or preferences. In domains where opponents are similar, these distributions tend to be more correlated.

by the average of each $u_i, i \in A'$ (since, given that all opponents in A' will accept the offer, the order in which the opponents accept them is equally likely) written formally as $f(\vec{u}, A') = \sum_{i \in A'} \frac{u_i}{|A'|}$.

Given this, we find the set of best values, $\vec{u}^*$, to offer to the opponents by maximising the expected utility. Formally:

$$\vec{u}^* = \text{argmax}_{\vec{u} \in [0,1]^{|A|}} EU_{\text{offer}}(\vec{u}, t^*) \tag{5.7}$$

Since the EU_{offer} function is nonlinear, we use a nonlinear optimisation package (specifically, the *Ipopt* interior point optimizer (Wächter and Biegler, 2006), using the HSL mathematical library (HSL, 2011)) to find the solution to Equation 5.7.

5.4 Handling Decommitment

The scenario that we introduced in Chapter 1 allowed a negotiating party to decommit from an agreement that they have previously formed, by paying a decommitment penalty. Therefore, our agent needs a method to decide when to decommit from such agreements.

In order for the agent to benefit from a decommitment, the value of the new agreement needs to be greater than that of the currently held agreement, plus the decommitment penalty. Formally:

$$u_{\text{new}} > u_{\text{existing}} + \rho \tag{5.8}$$

where u_{new} is the utility of the new agreement, u_{existing} is the utility of the existing agreement, and ρ is the decommitment penalty.

The benefit b can then be defined as:

$$b = u_{\text{new}} - u_{\text{existing}} - \rho \tag{5.9}$$

However, if the benefit obtained by accepting the new offer over the existing agreement is very small, it may not be rational for the agent to accept it. In more detail, the acceptance of an offer from an opponent leads to the termination of agreements with that opponent. Therefore, the number of negotiation partners is reduced by one, and a potential opportunity to reach an even better agreement is lost. Consequently, our agent will only accept a subsequent agreement if the benefit b is reasonably large.

Specifically, our agent only accepts subsequent agreements if:

$$b > (u_{\text{existing}} + \rho) \cdot \iota \tag{5.10}$$

where $\iota > 0$ is a factor which affects the size of the desired benefit. The value of ι should be small enough such that there can be offers which satisfy the above equation. For example, if $\iota = 0.2$, $\rho = 0.1$ and there is an existing agreement with utility $u_{\text{existing}} = 0.8$, the agent will only accept subsequent agreements if they have a utility greater than 1.08 (which of course, is not possible). Therefore, if ι is set to a value which is too large, there may be no possible subsequent agreements, and consequently, no decommitment would occur. We set $\iota = 0.1$ such that our agent has a desired benefit of at least 10% of the value of the existing offer plus the decommitment penalty. This means that only when $u_{\text{existing}} > 0.81$ (and $\rho = 0.1$) will our agent no longer be able to accept any subsequent agreements.[3]

5.5 Our Many-to-Many Negotiation Agent

We have developed a negotiating agent for use in concurrent negotiation settings, using the strategies described in Sections 5.2 to 5.4. We refer to this strategy as *IAMconcurrentHaggler*.

In common with our one-to-one negotiation agents, *IAMconcurrentHaggler* is implemented using the Java programming language, under the framework provided by the Genius platform. It extends the *IAMhaggler2012* agent and therefore uses *Commons-Math: The Apache Commons Mathematics Library*[4], *JAMA, A Java Matrix Package*[5] and the *Gaussian Process Regression for Java* library[6]. Furthermore, as a solver, it uses the *Ipopt* interior point optimizer (Wächter and Biegler, 2006), using the HSL mathematical library (HSL, 2011).

5.6 Summary

In this chapter, we have described the negotiating agent that we have designed for use in concurrent negotiation environments.

[3]Note that the actual value of $\iota = 0.1$ is somewhat arbitrary and other similar values do not significantly affect the performance from our agent.

[4]`http://commons.apache.org/math/`

[5]`http://math.nist.gov/javanumerics/jama/`

[6]`https://forge.ecs.soton.ac.uk/projects/gp4j/`

Specifically, we have contributed the following to the literature on automated negotiation:

- We have extended our one-to-one negotiation strategy to support the coordination of multiple, concurrent negotiations (Requirement 8) in which each participant aims to reach a single agreement, and decommitment of agreements is allowed, through payment of a penalty (Requirement 9).

Furthermore, against the requirements set out in Section 1.2, we have designed an agent which, in addition to meeting the same requirements as the agents detailed in Chapter 3, also:

- is able to effectively coordinate multiple concurrent negotiations with a range of opponents in order to reach effective outcomes. (Requirement 8)
- will decommit from agreements when it is beneficial for them to do so in order to improve their utility, considering the benefit of the new agreement over the decommitment penalty. (Requirement 9)

In the following chapter, we will evaluate the performance of our agent in order to show that it outperforms an existing state-of-the-art strategy for coordinating multiple negotiation threads (Nguyen and Jennings, 2005), in a range of scenarios.

Chapter 6

Evaluation of Many-to-Many Negotiation Agents

This chapter evaluates the performance of our many-to-many negotiation agent, which is designed to coordinate multiple concurrent negotiations (Requirement 8) in settings where decommitment is allowed (Requirement 9). We begin by introducing the scenarios (Section 6.1) and strategies (Section 6.2) that we consider in the evaluation. In a many-to-many negotiation, each party can have a different preference profile and use a different strategy. In order to reduce the amount of computation required to complete the evaluation, whilst continuing to evaluate a variety of settings, we analyse negotiations in which all opponents have the same preferences but use different strategies (Section 6.3) and then we analyse negotiations in which each opponent has different preferences but uses the same strategy (Section 6.4). Finally, we summarise the chapter (Section 6.5).

6.1 Evaluation Scenarios

In a many-to-many negotiation between $|A|$ parties, a scenario consists of $|A|$ preference profiles, each of which may be different. Furthermore, in the many-to-many negotiations that we consider, each party belongs to one of two classes, with each party aiming to reach an agreement with one member of the other class. For example, in the car sales scenario introduced in Chapter 1, the two classes represent the buyers and the sellers. The buyers are able to negotiate with all of the sellers, and vice-versa, but no agent negotiates with another agent of the same class.

Due to the many-to-many nature of these negotiation environments, to fully simulate them in our experiments requires considerable computation. In more detail, if there are a total of $|A|$ parties, with half of them in each class, there will be a total of $\frac{|A|}{2}$ agents, each negotiating with $\frac{|A|}{2}$ opponents, leading to $\left(\frac{|A|}{2}\right)^2$ negotiation pairs. However, to complete our analysis, we only require the utility of one of the $|A|$ negotiation parties. Therefore, we use the following approach in order to reduce the amount of computation required, without significantly changing the nature of the negotiation environment.

In our experiments, rather than simulating the competitors (i.e. those agents which belong to the same class) of our agent (or the alternatives that we introduce in Section 6.2.1), we represent them by a break-off function which affects our opponents. Taking this approach removes the need to simulate agents which do not have a direct effect on the performance of the agents we wish to compare. In more detail, if we are testing the performance of agent p which is in class 1, the only agents that it negotiates with are its opponents in class 2. The most significant effect that the competitors (who are in class 1 and do not negotiate directly with agent p) can have on agent p is to reach agreement with one of the opponents, thereby causing the opponent to leave the negotiation. From the perspective of agent p, this can be approximated by introducing a break-off function to the opponents in class 2, representing the agreement between an opponent and a competitor. In more detail, we model the probability of break-off using a time-invariant function. That is, at any time in the negotiation, the break-off probability during a future time period is given by a function which depends only on the length of that future period. We achieve this by using an exponential function to calculate the probability that an opponent continues to negotiate. Furthermore, we assume that all opponents have the same probability. Specifically, the continuation probability for a given period is given by:

$$\forall i \in Q, P_{c,i}(t_a, t_b) = \alpha^{t_b - t_a} \tag{6.1}$$

where t_a, $t_b > t_a$ are respectively the start and end of the period, and α is a constant which determines the rate of break-off. In our experiments, we set $\alpha = 1/|Q|$, where $|Q|$ is the total number of opponents. This ensures that, on average, there will be one agent remaining in the negotiation by the deadline, and is representative of an environment in which the number of negotiation parties in each class is equal.

By using the above approach to reduce the number of negotiation parties which need to be simulated, our negotiations contain $|Q| + 1$ agents which need to be simulated and therefore require preference profiles. Since one-to-one negotiations contain only 2 parties (and therefore 2 preference profiles), the scenarios used in Section 4.1 are unsuitable for a full evaluation of many-to-many negotiations. Therefore, we propose some additional scenarios, as follows.

Name	Number of issues	Number of values for each issue	Number of potential outcomes
Camera	6	3,3,4,4,5,5	3,600
ADG	6	5,5,5,5,5,5	15,625
Travel	7	4,4,5,6,7,7,8	188,160

TABLE 6.1: Characteristics of different scenario types.

Many of these scenarios used for one-to-one negotiation are not very competitive, and it is often easy for the agents to reach agreements with a high utility (for both sides), even when using a very simple strategy. In a many-to-many negotiation setting, reaching agreements with a high utility becomes even easier since, with a range of opponents, it only takes one weak (concessive) opponent to allow any strategy to reach a good agreement. As a result, such scenarios fail to offer sufficient challenge in a concurrent negotiation setting. To address this shortcoming, we generate a range of strictly opposing preference profiles. That is, the utility functions of any pair of negotiating agents, a and b (where a and b come from different classes), are such that:

$$\forall i \in I, \forall v_{i,x}, v_{i,y} \in V_i, U_{a,i}(v_{i,x}) \leq U_{a,i}(v_{i,y}) \Leftrightarrow U_{b,i}(v_{i,x}) \geq U_{b,i}(v_{i,y}) \tag{6.2}$$

where $v_{i,x}$ and $v_{i,y}$ are a pair of values allowed for issue i, $U_{a,i}(\cdot)$ is agent a's utility function for issue i, and $U_{b,i}(\cdot)$ is agent b's utility function for issue i.

To generate a variety of scenarios, we choose the values for each issue by sampling from a uniform distribution, and sort them such that the strict opposition constraint in Equation 6.2 is satisfied. We then normalise the values by dividing each one by the value of the greatest value, such that the greatest value is normalised to 1. Furthermore, the weights for each issue are also sampled from a uniform distribution, normalised such that they sum to one. Since we can generate any number of scenarios using this approach, we refer to the underlying characteristics (the number of issues and the number of values taken by those issues) as the *scenario type*. We consider three scenario types with large outcome spaces, based on those used in the competition. Specifically they are based on the *Camera*, *ADG* and *Travel* scenarios. The details of the scenario types are given in Table 6.1.

To ensure that we tested with a range of preference profile pairs with different levels of competitiveness, for each scenario type, we generated a single profile for one class and over 100 profiles for the opposing class. We then ranked these profile pairs according to their competitiveness, and we selected 7 pairs spread evenly throughout this sequence of profile pairs.

The competitiveness of each of these pairings, including the plots of their outcome spaces are shown in Table 6.2. The *Camera* scenario type is the least competitive, due to it having the lowest number of values for each issue, combined with the approach we use to generate the utility functions. Specifically, if issue 1 takes one of 3 values, and $U_{a,1}(v_{1,x}) = 1$, on average $U_{b,1}(v_{1,x}) = \frac{1}{3}$. In contrast, if issue 1 takes one of 8 values, $U_{a,1}(v_{1,x}) = 1$, on average $U_{b,1}(v_{1,x}) = \frac{1}{8}$, resulting in a much more competitive outcome space. At the other end of the scale, the *ADG* scenario type is the most competitive (but only slightly more so than the *Travel* scenario type).

Finally, in order to ensure that decommitment is a viable option for the participants, but is not completely free, we set $D = 0.1$. Moreover, as in the one-to-one negotiations we have considered in previous chapters, in each negotiation, there is a deadline of 3 minutes, which is common to all participants.

6.2 Evaluation Strategies

We now consider the strategies that are used in our experiments, in addition to our own strategy, *IAMconcurrentHaggler2012*. In Section 6.2.1 we detail two alternative strategies that can be used in place of ours. We use these two strategies as benchmarks, and therefore we compare their performance against the performance of *IAMconcurrentHaggler2012*. In our experiments, from the perspective of an agent using one of those three strategies, each of their opponents use one of the strategies detailed in Section 6.2.2.

6.2.1 Benchmark Strategies

We test our agent by comparing it against a state-of-the-art agent, designed for multiple concurrent negotiations (described in Section 6.2.1.1) and a very simple agent (described in Section 6.2.1.2) as benchmarks.

6.2.1.1 NguyenAgent

As the state-of-the-art benchmark agent, we use the strategy developed by Nguyen and Jennings (2005), which we introduced in Section 2.4.1. A limitation of this strategy, which we refer to as *NguyenAgent*, is that it requires prior knowledge about the payoffs of various strategies against different opponent classes (i.e., tough, linear, and conceder). To determine these values in a principled manner, we used the results from a set of negotiations between simple time-dependent strategies, in a bi-lateral negotiation setting. In more detail, we ran one-to-one negotiations between variants of a simple time-dependent conceder agent (either, tough ($\beta = 0.5$), linear ($\beta = 1.0$) or conceder ($\beta = 2.0$)), in all ANAC2011 scenarios, averaging the results across those scenarios in order to produce

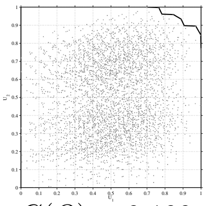
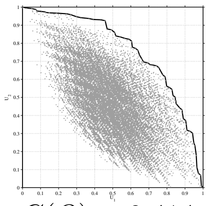
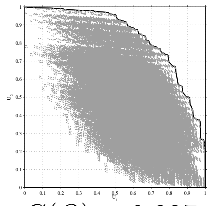
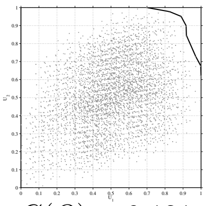
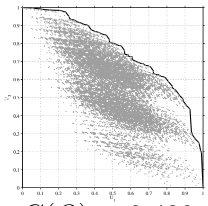
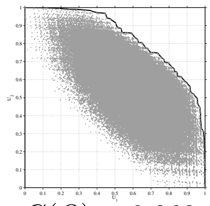
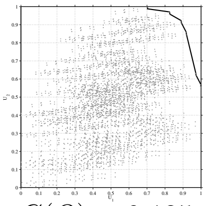
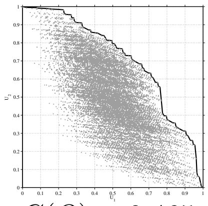
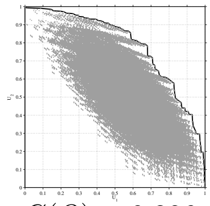
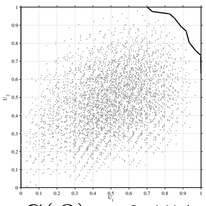
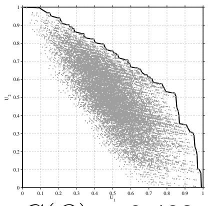
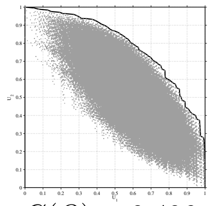
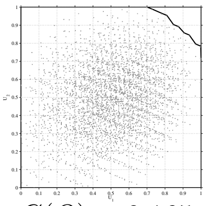
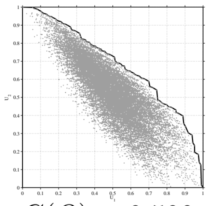
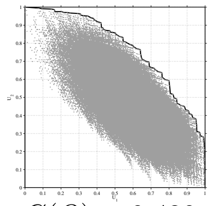
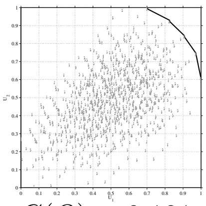
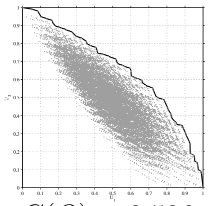
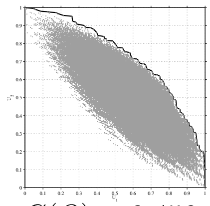
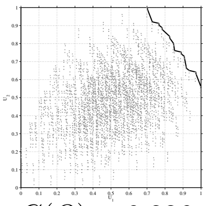
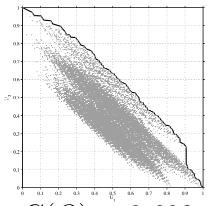
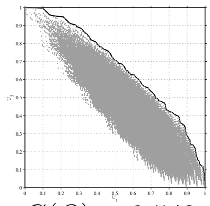
Camera	ADG	Travel
 $C(O) = 0.109$	 $C(O) = 0.414$	 $C(O) = 0.337$
 $C(O) = 0.121$	 $C(O) = 0.439$	 $C(O) = 0.363$
 $C(O) = 0.135$	 $C(O) = 0.465$	 $C(O) = 0.386$
 $C(O) = 0.154$	 $C(O) = 0.482$	 $C(O) = 0.406$
 $C(O) = 0.165$	 $C(O) = 0.509$	 $C(O) = 0.430$
 $C(O) = 0.181$	 $C(O) = 0.526$	 $C(O) = 0.456$
 $C(O) = 0.236$	 $C(O) = 0.603$	 $C(O) = 0.542$

TABLE 6.2: Competitiveness, $C(O)$ of different pairs of preference profile for each scenario, ordered from least to most competitive, showing outcome spaces. $C(O)$ is defined as the minimum distance from a point in the outcome space, O, to the point which represents complete satisfaction (that is, the point at which each agent achieves a utility of 1). The Pareto frontier of each profile pair is displayed as a solid line.

β	Opponent	
	Conceder	Non-conceder
2.0	0.366	0.130
1.0	0.510	0.120
0.5	0.525	0.111

Table 6.3: Payoff matrix used by *NguyenAgent* against 7 opponents, including effect of break-off.

the payoff matrices required by Nguyen's strategy. Specifically, the payoff matrix shows the utility that *NguyenAgent* will expect to achieve against a specific opponent if it chooses a tough, linear or conceder approach, depending on whether that opponent is a conceder or a non-conceder. Included in the payoff matrix is the effect of break-off, and therefore a matrix is produced for each different number of opponents (given that this number affects the average time of break-off and therefore the expected payoff). Table 6.3 shows the payoff matrix for *NguyenAgent* against a single opponent when the negotiation consists of a total of 7 opponents. Unsurprisingly, it shows that the expected utility is likely to be considerably higher if the opponent is a conceder than if it is a non-conceder. In terms of the response that *NguyenAgent* should take, it shows that the best response to a conceder is to use the tough ($\beta = 0.5$) approach, whereas against a non-conceder, the best response is to use the conceder ($\beta = 2.0$) approach.

We use the 'greedy' version of Nguyen's strategy, which, once an agreement has been reached, forms subsequent agreements which lead to an increase in the utility after considering the decommitment penalty. That is:

$$u_{\text{new}} > u_{\text{existing}} + \rho \tag{6.3}$$

where u_{new} is the utility of the new agreement, u_{existing} is the utility of the existing agreement, and ρ is the decommitment penalty.

6.2.1.2 RandomAgent

As an additional benchmark, we developed a simple agent, which we refer to as *RandomAgent*. This agent makes random offers above a fixed threshold which is chosen randomly from a uniform distribution over the range $[0, 1]$ in each negotiation session. After an agreement has been reached, this threshold is increased to the value of that agreement, plus the decommitment penalty. This ensures that any subsequent agreements lead to an improvement in the utility after considering the decommitment penalty.

Having introduced the agents which we compare our agent to, we now consider the strategies that their negotiation opponents use.

6.2.2 Opponent Strategies

As introduced in Section 6.1, in our experiments, we do not simulate the competitors (i.e. those agents which belong to the same class) of our agent (or the alternatives that we introduce in Section 6.2.1), but rather, we represent those competitors by a break-off function which affects our opponents. The main reason for this is as an approximation, in order to reduce the amount of computation required to perform the evaluation. However, it also provides us with a further advantage, as follows. Since the opponents negotiate with only one strategy, we can generate opponent strategies by making some minor adaptations to each of the large set of state-of-the-art, independently developed negotiation agents designed for one-to-one negotiation (as used in our evaluation in Chapter 4).

Specifically, in order to adapt these existing agents for the many-to-many protocol, they need to be capable of sending CONFIRM messages, and they need to represent agreement with a competitor through break-off. Since the only rational reason not to confirm an acceptance is if the agent has already reached another agreement, it is straightforward to add CONFIRM message functionality to the existing agents. Furthermore, the break-off is modelled by setting a time of break-off according to the probability of break-off function in Equation 6.1. Specifically, the time of break-off, t_{b}, is set as:

$$t_{\mathrm{b}} = \frac{\log(x)}{\log\left(\frac{1}{|Q|}\right)} \tag{6.4}$$

where x is a random variable drawn from a uniform distribution, $\mathcal{U}(0,1)$ over the range $[0,1]$, and $|Q|$ is the number of opponents.

Therefore, in this evaluation, we use all of the strategies in Section 4.5 (excluding our own) as opponent strategies, adapting them for our many-to-many setting by adding the break-off and confirmation of agreement features discussed above.

6.3 All Opponents have the Same Preferences

In this section, we consider negotiations in which all opponents have the same preference profile, but with each opponent using a different strategy (from the set of opponent strategies used in Chapter 4). We repeat the negotiations for each of the seven preference profile pairs in each scenario.

For each of the three agents (*IAMconcurrentHaggler*, *NguyenAgent*, and *RandomAgent*), we run experiments with different numbers of opponents. Each experiment consists

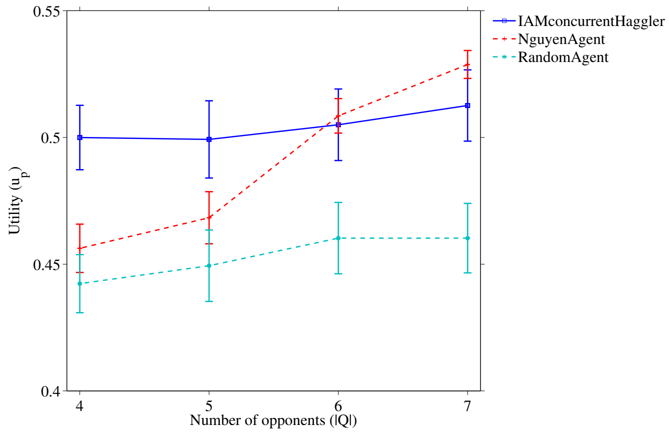
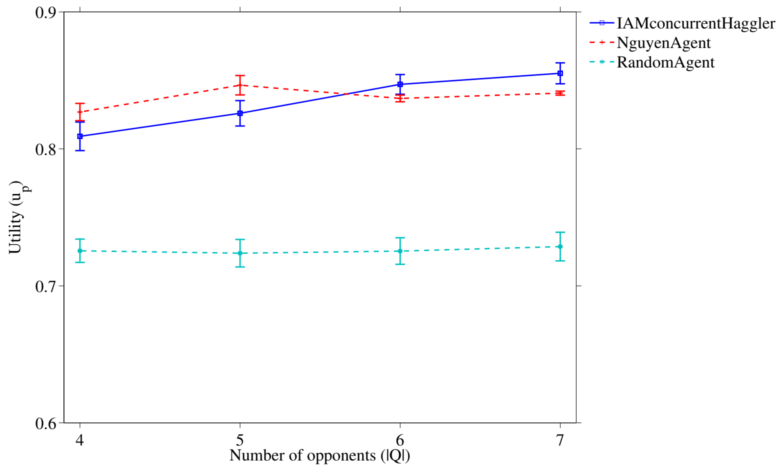


FIGURE 6.1: Average utility of the concurrent negotiation strategies in negotiations where each opponent uses the same preferences, but a different strategy, according to the number of opponents, averaged over all scenario types. Error bars show the 95% confidence intervals.

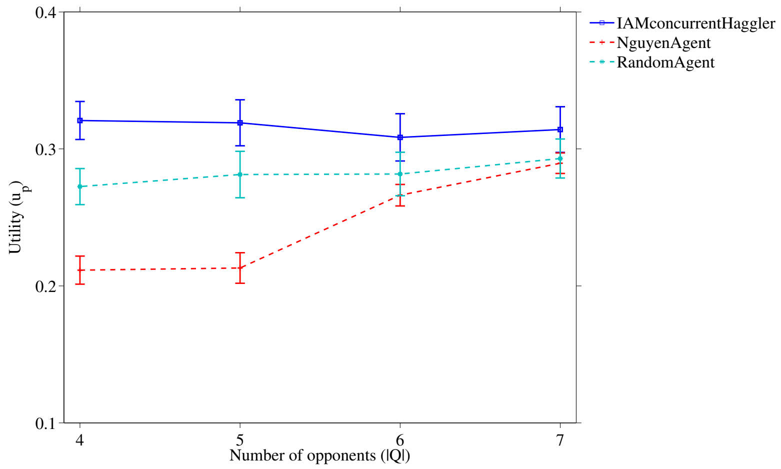
of up to 35 different negotiations per scenario type. In each negotiation, any opponent strategy from the ANAC competition appears at most once. At the same time, we select the set of opponent strategies in a particular negotiation such that each combination is equally represented within the experiment. For example, since we have 7 different opponents, if $|Q| = 4$, there are 35 different combinations, each run 4 times (in order to obtain statistically significant results). If $|Q| = 7$, then each opponent appears in all negotiations, and we repeat the negotiation 84 times.

The results of these experiments are shown in Figure 6.1, averaged over all scenarios. It shows that, for $|Q| \in \{4, 5\}$, *IAMconcurrentHaggler2012* significantly outperforms *NguyenAgent*, and for $|Q| \in \{6, 7\}$ the performance of the two agents are not significantly different to each other. Overall, on average *IAMconcurrentHaggler2012* outperforms *NguyenAgent* by 2.8%. Furthermore, for all values of $|Q| \in \{4, 5, 6, 7\}$, *IAMconcurrentHaggler2012* significantly outperforms the random benchmark (by an average of 11%).

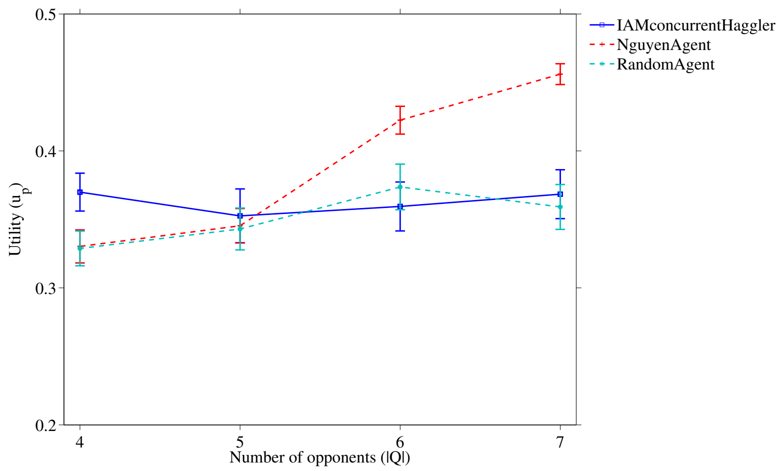
We now consider each individual scenario type: *Camera*, *ADG* and *Travel*. In the smallest, and least competitive scenario type (*Camera*) (Figure 6.2(a)), we observe that, in general, the performances of *NguyenAgent* and *IAMconcurrentHaggler2012* are so similar that the difference between them is not statistically significant, although they both significantly outperform *RandomAgent* (each by 15%). The reason that the performance of the two more advanced strategies is so similar is that, in such an uncompetitive scenario type, it is easy for any well designed agent to reach good agreements. In the most competitive scenario type (*ADG*) (Figure 6.2(b)), *IAMconcurrentHaggler2012* significantly outperforms *NguyenAgent*, achieving an average utility 29% higher. In the largest scenario type (*Travel*) (Figure 6.2(c)), *IAMconcurrentHaggler2012* is significantly



(a) *Camera* scenario type.



(b) *ADG* scenario type.



(c) *Travel* scenario type.

FIGURE 6.2: Average utility of the concurrent negotiation strategies in negotiations where each opponent uses the same preferences but a different strategy, according to the number of opponents, in individual scenario types. Error bars show the 95% confidence intervals.

outperformed by *NguyenAgent*, which achieves an average utility 7% higher, with the difference being greatest for larger values of $|Q|$. It should be noted that, as the number of opponents increases, there is more opportunity for agreement, particularly when those opponents are all using a different strategy (as is the case in these experiments). In more detail, each agent is only aiming to reach agreement with a single opponent, and if just one of those opponents is particularly concessive it is very easy for an agent to obtain a high utility. If the opponents are all using different strategies, the more opponents there are, the higher the probability of encountering such a concessive opponent and therefore, it becomes easier to obtain a higher utility. *NguyenAgent* takes more of an advantage of this (compared to *IAMconcurrentHaggler2012*) by using a less concessive strategy.

6.4 Each Opponent has Different Preferences

In this section, we consider negotiations in which each opponent has a different preference profile, but each opponent uses the same strategy. Therefore, we repeat the negotiations for each of the 7 opponent strategies.

For each of the three agents (*IAMconcurrentHaggler2012*, *NguyenAgent*, and *RandomAgent*), we run experiments with different numbers of opponents. Each experiment consists of up to 35 different negotiations per scenario type. In each negotiation, only one of the opponent strategies from the ANAC competition appears. At the same time, we select the set of preference profiles in a particular negotiation such that each combination is equally represented within the experiment. For example, since we have 7 different profiles, if $|Q| = 4$, there are 35 different combinations, each run 4 times (in order to obtain statistically significant results). If $|Q| = 7$, then each profile appears in all negotiations, and we repeat the negotiation 84 times.

The results of these experiments are shown in Figure 6.3, averaged over all scenarios. It shows that, for $|Q| \in \{4, 5\}$, *IAMconcurrentHaggler2012* significantly outperforms *NguyenAgent*, and for $|Q| \in \{6, 7\}$ the performance of the two agents are not significantly different to each other. As the number of opponents increases, the probability of an individual opponent breaking off the negotiation increases and therefore *IAMconcurrentHagger2012* tends to become more concessive against each opponent. Overall, on average *IAMconcurrentHaggler2012* outperforms *NguyenAgent* by 6.6%. Furthermore, for all values of $|Q| \in \{4, 5, 6, 7\}$, *IAMconcurrentHaggler2012* significantly outperforms the random benchmark (by an average of 6.5%).

We now consider each individual scenario type: *Camera*, *ADG* and *Travel*. In the smallest, and least competitive scenario type (*Camera*) (Figure 6.2(a)), we see that, in general

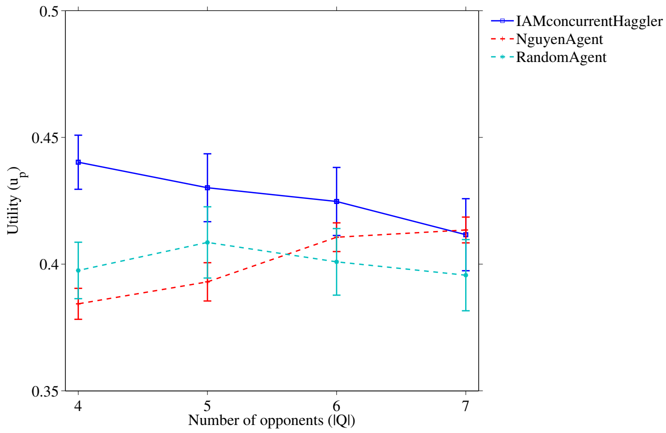
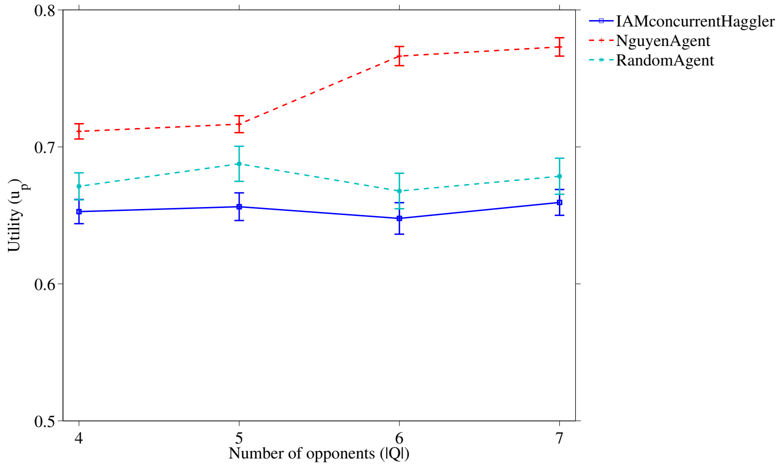


FIGURE 6.3: Average utility of the concurrent negotiation strategies in negotiations where each opponent uses different preferences, but the same strategy, according to the number of opponents, averaged over all scenario types. Error bars show the 95% confidence intervals.

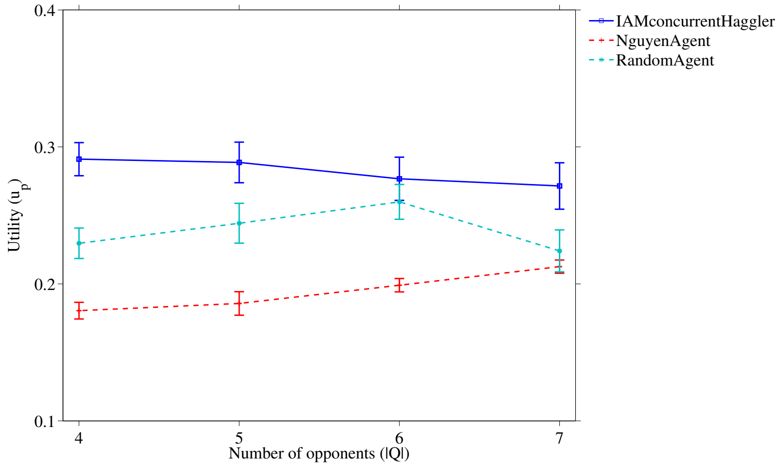
NguyenAgent achieves a higher average utility than that of *IAMconcurrentHaggler* (by 13%). In the most competitive scenario type (*ADG*) (Figure 6.2(b)), *IAMconcurrentHaggler2012* significantly outperforms *NguyenAgent*, for all values of $|Q|$, achieving an average utility 45% higher. In the largest scenario type (*Travel*) (Figure 6.2(c)), *IAMconcurrentHaggler2012* also significantly outperforms *NguyenAgent*, for all values of $|Q|$, achieving an average utility 30% higher. Increasing the number of opponents causes the expected break-off time for a single opponent to be earlier. As a result, *IAMconcurrentHaggler2012* takes a more concessive approach. Although the preferences differed in these experiments, the difference between them was relatively small. Since the strategies used by the opponents were identical, it is likely that similar behaviour would have been observed despite the difference in preferences. Therefore, our agent has conceded more than necessary, as $|Q|$ increases, resulting in a slight decreasing trend in the utility achieved, where an increasing one would otherwise have been expected.

6.5 Summary

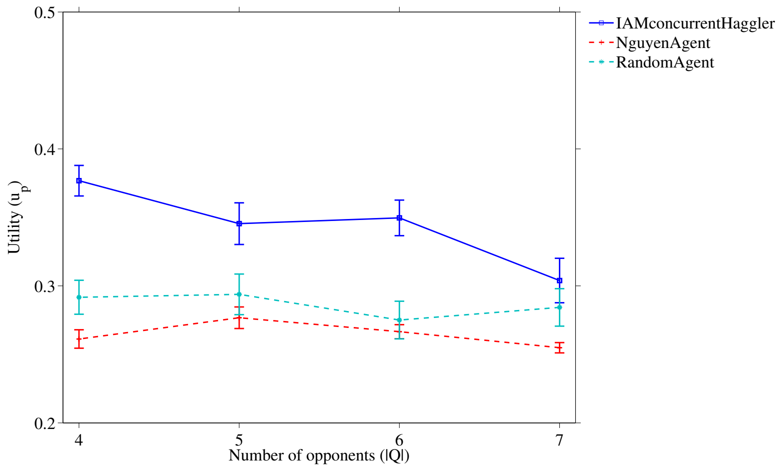
In this chapter we have shown the performance of our many-to-many negotiation agent compared to that of two benchmark strategies. Specifically, we have shown that our strategy is effective at coordinating multiple concurrent negotiations (Requirement 8) in settings where decommitment is allowed (Requirement 9), outperforming the state-of-the-art benchmark across a range of different scenario types (on average, by 4.7%). It is a particularly strong strategy for highly competitive scenario types, such as the *ADG* and *Travel* types, where, in some cases (*ADG* scenario type with 4 opponents each



(a) *Camera* scenario type.



(b) *ADG* scenario type.



(c) *Travel* scenario type.

FIGURE 6.4: Average utility of the concurrent negotiation strategies in negotiations where each opponent uses different preferences, but the same strategy, according to the number of opponents, in individual scenario types. Error bars show the 95% confidence intervals.

having a different profile), it outperforms *NguyenAgent* by as much as 61%. In other cases, it is not much better, or even may be worse than the state-of-the-art benchmark.

Chapter 7

Conclusions and Future Work

This thesis begins by introducing the challenges associated with automated negotiation in complex environments (in Chapter 1) along with related work (in Chapter 2). It then describes (in Chapter 3) the negotiating agents that we have developed, which address some of the limitations of the existing approaches, and meet the research requirements that we set out in Chapter 1. Specifically, the agents use a decentralised approach (Requirement 1) to negotiate over multiple issues (Requirement 4) with an unknown ordering (Requirement 5). Furthermore, in our environment, negotiation occurs in real time (Requirement 6), with a real-time deadline and discounting factor, as opposed to much of the literature on bilateral negotiation, where the time constraints depend on the number of interactions, not the actual elapsed time, and therefore any deliberation time by the agent is not taken into account. In terms of performance, the main challenge was to design a strategy that could achieve a high utility in negotiations where the preferences and the behaviour of the opponents are unknown (Requirement 3). Furthermore, the strategy that we designed needed to reach efficient agreements (Requirement 2), and be computationally tractable (Requirement 7). To this end, we developed two concession strategies as follows. The first uses a fast, least squares regression approach to predict the future concession of the opponent, which can be repeated after each offer. The second, more advanced strategy uses a slower, Gaussian process regression technique, which, in addition to the prediction, also provides a measure of the confidence in that prediction. It then uses this information in order to set its concession as a best response to the opponent's behaviour.

As part of our evaluation (Chapter 4), we have compared the performance of our agents against the performance of those used in the Automated Negotiation Agent Competitions (ANAC). Furthermore, we also use the scenarios from ANAC 2012 in our evaluation. These scenarios and agents are a good representation of the state-of-the-art, and they form an independently developed, varied set of opponents and settings which are ideal for benchmarking our strategies against. Overall, in a tournament consisting of the top 8 strategies from ANAC 2012, taking the scores averaged over all scenarios, our most advanced one-to-one negotiation agent, *IAMhaggler2012* finishes in 5th place. Despite this relatively low ranking, the utility it achieves is still very close to that of the winning agent. Specifically, its utility is 96% of that of the winning agent, *AgentLG*, whereas the lowest scoring agent, *AgentMR* only achieves 52%. However, there are a number of situations in which *IAMhaggler2012* outperforms all of the other strategies. In more detail, in the largest scenario, with discounting, its average utility in a tournament setting was 33% higher than the agent with the second highest average utility. Furthermore, *IAMhaggler2012* achieves the highest self-play utility (that is, when both parties use the same strategy). We have also shown that the agreements which are reached are approximately Pareto efficient (Requirement 2), and that our strategy is computationally tractable (Requirement 7). Additionally, we have applied an empirical game theoretic technique to analyse the results of a set of negotiation tournaments. By using this technique, we have shown that, in certain scenarios (particularly those in which are highly discounted ($\delta = 0.5$)), for the best performing subset of the strategies of the agents present in the ANAC 2012 final, there is no incentive for any of the agents to switch from using our *IAMhaggler2012* strategy, to using another of those strategies.

Furthermore, in order to address our requirements for a strategy which can coordinate multiple concurrent negotiations with a range of opponents (Requirement 8) in an environment where decommitment is allowed (Requirement 9), we develop a further strategy (Chapter 5). Specifically, this agent, *IAMconcurrentHaggler2012*, uses Gaussian process regression to predict the future concession of each opponent. It uses these predictions to coordinate the concession against all of its opponents. Once an agreement has been reached, the agent continues negotiating in an attempt to reach a better agreement (after payment of a decommitment penalty) with a different opponent.

Finally, in our evaluation of *IAMconcurrentHaggler* (Chapter 6), we have shown how, in negotiations with only a small number of opponents (4 or 5), our agent achieves, on average, higher utilities than both the state-of-the-art and random benchmarks. Although in the least competitive scenario type, *IAMconcurrentHaggler* is slightly outperformed by the state-of-the-art benchmark, in the highly competitive scenario types, our strategy achieves a substantially higher average utility than both the state-of-the-art (by up to 45%) and random (by up to 29%) benchmarks.

Overall, the work in this thesis advances the state-of-the-art by proposing a novel, principled approach to concession in complex one-to-one negotiations. Whilst the strategy does not outperform all of the other strategies in all scenarios, it shows strong performance in highly discounted scenarios, in which it is desirable to reach outcomes without significant delay. Our extended strategy for many-to-many negotiation is one of the first to coordinate concession against a range of opponents, in such a complex environment. Our evaluation of this strategy shows that it outperforms an existing state-of-the-art many-to-many negotiation strategy in the most competitive scenarios. Furthermore, our evaluation shows how empirical game theoretic analysis can be used to consider the robustness of a strategy in a tournament setting, against a range of different opponents.

7.1 Future Work

While we have advanced the state-of-the-art in the development of our negotiation strategy, there are a number of ways in which this work could be extended. Therefore we now introduce some ideas for future work, beginning with a set of possible improvements to our model and concluding with more general ideas which address the interaction between negotiation agents and humans.

Consider Influence on Adaptive Opponents: In all of our strategies, the approach that our agent takes is to optimise its response to the behaviour of any opponents, assuming that those opponents use a fixed strategy which does not respond to our own behaviour. In practice this may not always be the case, since many of the strategies (including our own) adapt to the behaviour of the opponent. If our strategy can be extended to consider the effect that it can have on an opponent which adapts to our offers, then it can potentially be less adaptive, instead relying on the adaptiveness of its opponents, in order to reduce its concession and achieve a greater utility.

Consider Dependency between Similar Points in Time: Once our agent has determined the time, t^*, at which it expects the discounted utility of the opponent to be maximised, it aims to maximise its expected utility by considering only agreements that can be made at that time. In order to produce a computationally tractable approach, this possibility of reaching an agreement at times other than t^* is not included in our current model. However, in practice, if the agent fails to reach an agreement at that time, it will continue to negotiate and is still likely to be able to reach some form of agreement. Consequently, our strategy may be taking a more concessionary approach than necessary as it focuses on the need to reach agreement at t^*. In terms of addressing this limitation, it would be necessary to consider that the discounting factor will have a greater effect on any later agreement. Furthermore, the probability of acceptance at time $t + \epsilon$ is very similar to the probability of acceptance at time t, and this correlation would also need to be considered in an enhanced model. As part of this enhancement, the rate of offers need to be considered, as this rate gives an indication of the number of remaining offers before the deadline and therefore the number of opportunities for agreement.

Consider Dependency between Behaviour of Opponents in Many-to-Many Negotiations: Another assumption which is implicit in the design of our many-to-many negotiation strategy is that the behaviour of each opponent is independent of each other. However, if any of the opponents use similar strategies or preferences (as is the case in the negotiations we consider in Chapter 6), then there is likely to be a correlation between the behaviour observed from those opponents, which should be considered in order to more accurately predict the future concession of the opponents.

Applying Our Strategies to Negotiations against Humans: Another direction which could be taken in order to extend this work is to consider how our strategy could be adapted for use in negotiations where some of the opponents are human. In such a setting, any agreement with these human opponents would need to be made after much fewer rounds than is common in a set of negotiations where all participants are represented by agents. Furthermore, due to this limitation on the number of rounds, caused by to the increased time taken by a human opponent, the agent may also benefit by spending more time in computing each offer. Currently, our strategies take advantage of the high number of offers which can be made in an agent-only negotiation environment. As the number of offers becomes more limited, it becomes more important, at a given utility level, for the agent to propose the offers that are most likely to be accepted by the opponent. One way in which our strategy could be adapted to achieve this is to use a technique based on the one proposed by Hindriks and Tykhonov (2008), which uses Bayesian learning in order to model the utility functions of the opponents, and uses this model to search for Pareto-optimal offers. Lin and Kraus (2010) review a range of agents specifically designed for negotiations involving agents and humans, and their work provides important insights into the considerations necessary for such negotiation.

Communication between Agents and the Humans they Represent: Even in a negotiation environment where all of the negotiators are represented by agents, there are still a number of aspects to be considered regarding the interaction between each agent and the human it represents. These include the need for humans to express their preferences in the form of a utility function. Pommeranz et al. (2008) evaluate a range of different techniques for elicitation of preferences for use with automated negotiation.

It may also be desirable for the agent to provide feedback to the human to indicate why it behaves in a particular way, and to provide assurances that the agent is performing the best it can.

Appendix A

Scenarios

This appendix details all of the scenarios that were used in the evaluation sections. These scenarios were gathered from the 2010, 2011 and 2012 editions of the Automated Negotiating Agents Competition. In 2010, the scenarios were selected by the organisers. In 2011 and 2012, the scenarios were designed by the participants.

A.1 Airport Site Selection Scenario

Domain	Profile 1	Profile 2
Issue 1: Cost	$w_{1,1} = 0.5$	$w_{2,1} = 0.25$
$V_{1,1} =$ GreaterThanOrEqual5.2bn	$U_{1,1}(V_{1,1}) = 0.1$	$U_{2,1}(V_{1,1}) = 0.111$
$V_{1,2} =$ 5.0bn	$U_{1,1}(V_{1,2}) = 0.2$	$U_{2,1}(V_{1,2}) = 1$
$V_{1,3} =$ 4.8bn	$U_{1,1}(V_{1,3}) = 0.3$	$U_{2,1}(V_{1,3}) = 0.889$
$V_{1,4} =$ 4.6bn	$U_{1,1}(V_{1,4}) = 0.4$	$U_{2,1}(V_{1,4}) = 0.778$
$V_{1,5} =$ 4.4bn	$U_{1,1}(V_{1,5}) = 0.5$	$U_{2,1}(V_{1,5}) = 0.667$
$V_{1,6} =$ 4.2bn	$U_{1,1}(V_{1,6}) = 0.6$	$U_{2,1}(V_{1,6}) = 0.556$
$V_{1,7} =$ 4.0bn	$U_{1,1}(V_{1,7}) = 0.7$	$U_{2,1}(V_{1,7}) = 0.444$
$V_{1,8} =$ 3.8bn	$U_{1,1}(V_{1,8}) = 0.8$	$U_{2,1}(V_{1,8}) = 0.333$
$V_{1,9} =$ 3.6bn	$U_{1,1}(V_{1,9}) = 0.9$	$U_{2,1}(V_{1,9}) = 0.222$
$V_{1,10} =$ LessThanOrEqual3.4bn	$U_{1,1}(V_{1,10}) = 1$	$U_{2,1}(V_{1,10}) = 0.111$
Issue 2: Noise	$w_{1,2} = 0.25$	$w_{2,2} = 0.25$
$V_{2,1} =$ LessThan10000	$U_{1,2}(V_{2,1}) = 0.286$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2} = 10000$	$U_{1,2}(V_{2,2}) = 0.429$	$U_{2,2}(V_{2,2}) = 0.857$
$V_{2,3} = 20000$	$U_{1,2}(V_{2,3}) = 0.571$	$U_{2,2}(V_{2,3}) = 0.714$
$V_{2,4} = 30000$	$U_{1,2}(V_{2,4}) = 1$	$U_{2,2}(V_{2,4}) = 0.571$
$V_{2,5} = 40000$	$U_{1,2}(V_{2,5}) = 0.571$	$U_{2,2}(V_{2,5}) = 0.429$
$V_{2,6} = 50000$	$U_{1,2}(V_{2,6}) = 0.429$	$U_{2,2}(V_{2,6}) = 0.286$
$V_{2,7} =$ GreaterThan50000	$U_{1,2}(V_{2,7}) = 0.286$	$U_{2,2}(V_{2,7}) = 0.143$
Issue 3: AccidentLevelPerMillionPassengerMiles	$w_{1,3} = 0.25$	$w_{2,3} = 0.5$
$V_{3,1} =$ GreaterThanOrEqual0.1	$U_{1,3}(V_{3,1}) = 0$	$U_{2,3}(V_{3,1}) = 0$
$V_{3,2} = 0.08$	$U_{1,3}(V_{3,2}) = 0.2$	$U_{2,3}(V_{3,2}) = 0.2$
$V_{3,3} = 0.06$	$U_{1,3}(V_{3,3}) = 0.4$	$U_{2,3}(V_{3,3}) = 0.4$
$V_{3,4} = 0.04$	$U_{1,3}(V_{3,4}) = 0.6$	$U_{2,3}(V_{3,4}) = 0.6$
$V_{3,5} = 0.02$	$U_{1,3}(V_{3,5}) = 0.8$	$U_{2,3}(V_{3,5}) = 0.8$
$V_{3,6} =$ LessThan0.02	$U_{1,3}(V_{3,6}) = 1$	$U_{2,3}(V_{3,6}) = 1$

Table A.1: Airport Site Selection scenario specification.

Rank	Agent	Score
1-2	TheNegotiator Reloaded	0.623 ±0.002
2-4	IAMhaggler2012	0.617 ±0.002
2-4	OMACagent	0.617 ±0.003
1-6	CUHKAgent	0.613 ±0.013
4-5	AgentLG	0.604 ±0.005
5-6	Meta-Agent	0.593 ±0.007
7	BRAMAgent2	0.572 ±0.009
8	AgentMR	0.388 ±0.002

TABLE A.2: Scores in the Airport Site Selection scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

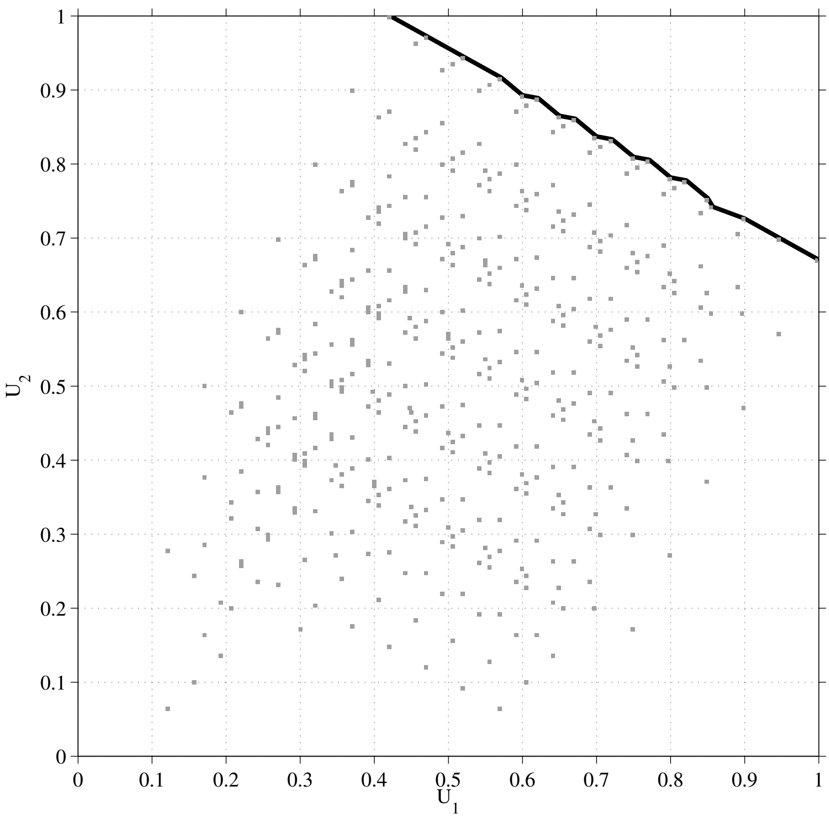


FIGURE A.1: Airport Site Selection scenario outcome space.

A.2 Barbecue Scenario

Domain	Profile 1	Profile 2
Issue 1: Meat	$w_{1,1} = 0.3$	$w_{2,1} = 0.3$
$V_{1,1}$ = BURGERS AND CHICKEN	$U_{1,1}(V_{1,1}) = 0.6$	$U_{2,1}(V_{1,1}) = 0.01$
$V_{1,2}$ = FISH	$U_{1,1}(V_{1,2}) = 0.5$	$U_{2,1}(V_{1,2}) = 0.8$
$V_{1,3}$ = LUXURY MEATS	$U_{1,1}(V_{1,3}) = 1$	$U_{2,1}(V_{1,3}) = 0.3$
$V_{1,4}$ = BIOLOGICAL	$U_{1,1}(V_{1,4}) = 0.7$	$U_{2,1}(V_{1,4}) = 0.5$
$V_{1,5}$ = VEGETARIAN	$U_{1,1}(V_{1,5}) = 0.1$	$U_{2,1}(V_{1,5}) = 1$
$V_{1,6}$ = NONE	$U_{1,1}(V_{1,6}) = 0.01$	$U_{2,1}(V_{1,6}) = 1$
Issue 2: Drinks	$w_{1,2} = 0.25$	$w_{2,2} = 0.1$
$V_{2,1}$ = NON-ALCOHOLIC	$U_{1,2}(V_{2,1}) = 0.1$	$U_{2,2}(V_{2,1}) = 0.6$
$V_{2,2}$ = BEER PLUS	$U_{1,2}(V_{2,2}) = 0.7$	$U_{2,2}(V_{2,2}) = 0.7$
$V_{2,3}$ = BIO-BEER	$U_{1,2}(V_{2,3}) = 0.1$	$U_{2,2}(V_{2,3}) = 1$
$V_{2,4}$ = LUXURY ALCOHOLIC	$U_{1,2}(V_{2,4}) = 1$	$U_{2,2}(V_{2,4}) = 0.9$
Issue 3: Location	$w_{1,3} = 0.25$	$w_{2,3} = 0.2$
$V_{3,1}$ = BALCONY	$U_{1,3}(V_{3,1}) = 0.3$	$U_{2,3}(V_{3,1}) = 0.3$
$V_{3,2}$ = WOODS	$U_{1,3}(V_{3,2}) = 0.5$	$U_{2,3}(V_{3,2}) = 1$
$V_{3,3}$ = PARK	$U_{1,3}(V_{3,3}) = 0.7$	$U_{2,3}(V_{3,3}) = 0.4$
$V_{3,4}$ = BEACH	$U_{1,3}(V_{3,4}) = 1$	$U_{2,3}(V_{3,4}) = 0.7$
Issue 4: Vegetables	$w_{1,4} = 0.1$	$w_{2,4} = 0.3$
$V_{4,1}$ = NONE	$U_{1,4}(V_{4,1}) = 0.8$	$U_{2,4}(V_{4,1}) = 0.01$
$V_{4,2}$ = CHEAP VEGGIES	$U_{1,4}(V_{4,2}) = 1$	$U_{2,4}(V_{4,2}) = 0.2$
$V_{4,3}$ = GOOD VEGGIES	$U_{1,4}(V_{4,3}) = 0.4$	$U_{2,4}(V_{4,3}) = 0.7$
$V_{4,4}$ = PREPARED VEGGIES	$U_{1,4}(V_{4,4}) = 0.6$	$U_{2,4}(V_{4,4}) = 0.6$
$V_{4,5}$ = SUPERVEGGIES	$U_{1,4}(V_{4,5}) = 0.7$	$U_{2,4}(V_{4,5}) = 1$
Issue 5: BBQ type	$w_{1,5} = 0.1$	$w_{2,5} = 0.1$
$V_{5,1}$ = DISPOSABLE	$U_{1,5}(V_{5,1}) = 0.9$	$U_{2,5}(V_{5,1}) = 0.1$
$V_{5,2}$ = NORMAL	$U_{1,5}(V_{5,2}) = 1$	$U_{2,5}(V_{5,2}) = 0.6$
$V_{5,3}$ = GAS	$U_{1,5}(V_{5,3}) = 0.8$	$U_{2,5}(V_{5,3}) = 1$

TABLE A.3: Barbecue scenario specification.

Rank	Agent	Score
1	AgentLG	0.644 ±0.004
2	OMACagent	0.636 ±0.004
3-4	TheNegotiator Reloaded	0.626 ±0.004
3-4	CUHKAgent	0.622 ±0.003
5-7	IAMhaggler2012	0.596 ±0.008
5-7	BRAMAgent2	0.593 ±0.008
5-7	Meta-Agent	0.593 ±0.010
8	AgentMR	0.188 ±0.000

TABLE A.4: Scores in the Barbecue scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}$, $u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

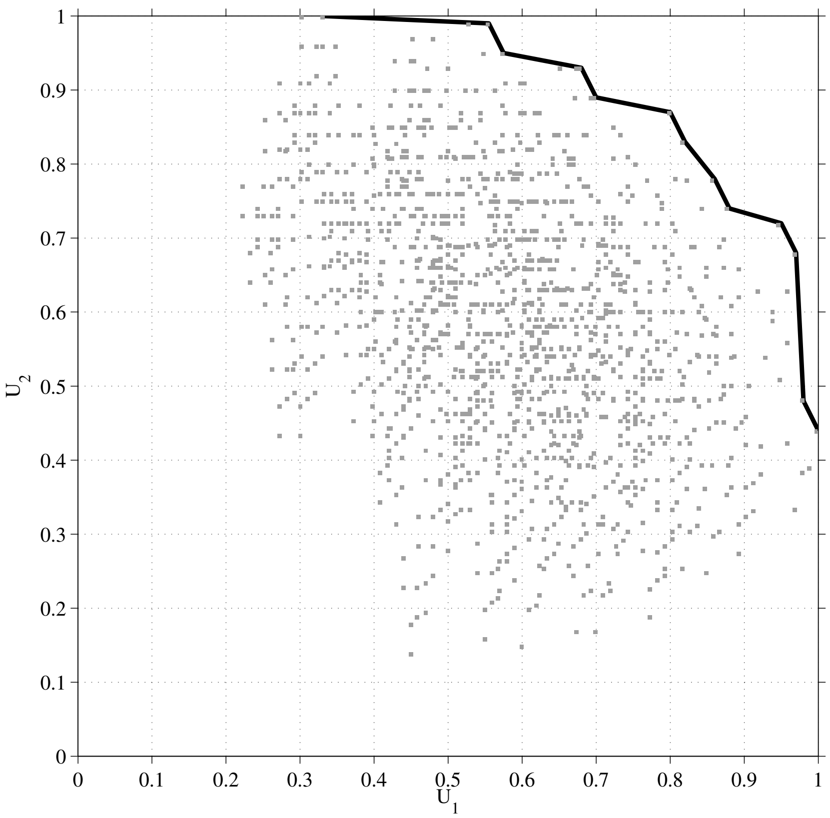


FIGURE A.2: Barbecue scenario outcome space.

A.3 Barter Scenario

Domain	Profile 1	Profile 2
Issue 1: price	$w_{1,1} = 0.36$	$w_{2,1} = 0.4$
$V_{1,1} = 3$GM	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.25$
$V_{1,2} = 4$GM	$U_{1,1}(V_{1,2}) = 0.75$	$U_{2,1}(V_{1,2}) = 0.5$
$V_{1,3} = 5$GM	$U_{1,1}(V_{1,3}) = 0.5$	$U_{2,1}(V_{1,3}) = 0.75$
$V_{1,4} = 6$GM	$U_{1,1}(V_{1,4}) = 0.25$	$U_{2,1}(V_{1,4}) = 1$
Issue 2: CookingOil	$w_{1,2} = 0.32$	$w_{2,2} = 0.4$
$V_{2,1} = 4$ML	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0.2$
$V_{2,2} = 5$ML	$U_{1,2}(V_{2,2}) = 0.8$	$U_{2,2}(V_{2,2}) = 0.4$
$V_{2,3} = 6$ML	$U_{1,2}(V_{2,3}) = 0.6$	$U_{2,2}(V_{2,3}) = 0.6$
$V_{2,4} = 7$ML	$U_{1,2}(V_{2,4}) = 0.4$	$U_{2,2}(V_{2,4}) = 0.8$
$V_{2,5} = 8$ML	$U_{1,2}(V_{2,5}) = 0.2$	$U_{2,2}(V_{2,5}) = 1$
Issue 3: Sugar	$w_{1,3} = 0.32$	$w_{2,3} = 0.2$
$V_{3,1} = 4$GM	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0.25$
$V_{3,2} = 6$GM	$U_{1,3}(V_{3,2}) = 0.75$	$U_{2,3}(V_{3,2}) = 0.5$
$V_{3,3} = 8$GM	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.75$
$V_{3,4} = 10$GM	$U_{1,3}(V_{3,4}) = 0.25$	$U_{2,3}(V_{3,4}) = 1$

Table A.5: Barter scenario specification.

Rank	Agent	Score
1-2	OMACagent	0.503 ±0.009
1-2	AgentLG	0.492 ±0.004
3	IAMhaggler2012	0.483 ±0.006
4-5	BRAMAgent2	0.471 ±0.004
4-5	CUHKAgent	0.468 ±0.001
6-7	TheNegotiator Reloaded	0.429 ±0.001
6-7	Meta-Agent	0.426 ±0.006
8	AgentMR	0.300 ±0.001

TABLE A.6: Scores in the Barter scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

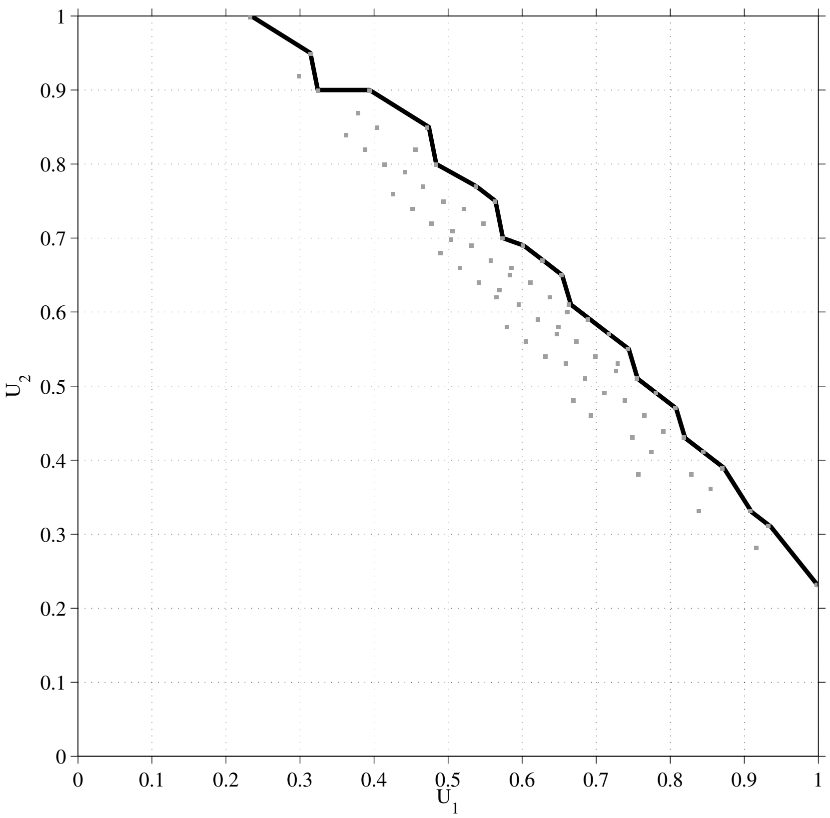


FIGURE A.3: Barter scenario outcome space.

A.4 Camera (2012) Scenario

Domain	Profile 1	Profile 2
Issue 1: Maker	$w_{1,1} = 0.34$	$w_{2,1} = 0.08$
$V_{1,1}$ = Canon	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.01$
$V_{1,2}$ = Nikon	$U_{1,1}(V_{1,2}) = 0.938$	$U_{2,1}(V_{1,2}) = 0.14$
$V_{1,3}$ = Pentax	$U_{1,1}(V_{1,3}) = 0.031$	$U_{2,1}(V_{1,3}) = 1$
$V_{1,4}$ = Sony	$U_{1,1}(V_{1,4}) = 0.078$	$U_{2,1}(V_{1,4}) = 0.6$
$V_{1,5}$ = Panasonic	$U_{1,1}(V_{1,5}) = 0.094$	$U_{2,1}(V_{1,5}) = 0.04$
Issue 2: Body	$w_{1,2} = 0.06$	$w_{2,2} = 0.09$
$V_{2,1}$ = Full size	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0.095$
$V_{2,2}$ = APS-C	$U_{1,2}(V_{2,2}) = 0.333$	$U_{2,2}(V_{2,2}) = 1$
$V_{2,3}$ = Micro Four Thirds	$U_{1,2}(V_{2,3}) = 0.133$	$U_{2,2}(V_{2,3}) = 0.381$
$V_{2,4}$ = compact	$U_{1,2}(V_{2,4}) = 0.067$	$U_{2,2}(V_{2,4}) = 0.032$
Issue 3: Lens	$w_{1,3} = 0.13$	$w_{2,3} = 0.34$
$V_{3,1}$ = High end model	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = Middle range model	$U_{1,3}(V_{3,2}) = 0.286$	$U_{2,3}(V_{3,2}) = 0.5$
$V_{3,3}$ = Entry model	$U_{1,3}(V_{3,3}) = 0.086$	$U_{2,3}(V_{3,3}) = 0.04$
Issue 4: Tripod	$w_{1,4} = 0.09$	$w_{2,4} = 0.26$
$V_{4,1}$ = GITZO	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0.313$
$V_{4,2}$ = Manfrotto	$U_{1,4}(V_{4,2}) = 0.476$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3}$ = Induro	$U_{1,4}(V_{4,3}) = 0.048$	$U_{2,4}(V_{4,3}) = 0.125$
Issue 5: Bag	$w_{1,5} = 0.11$	$w_{2,5} = 0.17$
$V_{5,1}$ = Domke	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0.2$
$V_{5,2}$ = Lowepro	$U_{1,5}(V_{5,2}) = 0.162$	$U_{2,5}(V_{5,2}) = 1$
$V_{5,3}$ = Tamrac	$U_{1,5}(V_{5,3}) = 0.103$	$U_{2,5}(V_{5,3}) = 0.1$
$V_{5,4}$ = National Geographic	$U_{1,5}(V_{5,4}) = 0.132$	$U_{2,5}(V_{5,4}) = 0.3$
$V_{5,5}$ = Artizan&Artist	$U_{1,5}(V_{5,5}) = 0.044$	$U_{2,5}(V_{5,5}) = 0.4$
Issue 6: Accessory	$w_{1,6} = 0.26$	$w_{2,6} = 0.07$
$V_{6,1}$ = Electronic Flash	$U_{1,6}(V_{6,1}) = 0.182$	$U_{2,6}(V_{6,1}) = 1$
$V_{6,2}$ = Battery Grip	$U_{1,6}(V_{6,2}) = 1$	$U_{2,6}(V_{6,2}) = 0.429$
$V_{6,3}$ = Memory	$U_{1,6}(V_{6,3}) = 0.045$	$U_{2,6}(V_{6,3}) = 0.571$
$V_{6,4}$ = Strap	$U_{1,6}(V_{6,4}) = 0.023$	$U_{2,6}(V_{6,4}) = 0.143$

Table A.7: Camera (2012) scenario specification.

Rank	Agent	Score
1-2	AgentLG	0.704 ±0.003
1-3	TheNegotiator Reloaded	0.694 ±0.007
2-3	CUHKAgent	0.690 ±0.002
4	OMACagent	0.680 ±0.003
5	Meta-Agent	0.661 ±0.004
6-7	IAMhaggler2012	0.643 ±0.005
6-7	BRAMAgent2	0.640 ±0.011
8	AgentMR	0.444 ±0.005

TABLE A.8: Scores in the Travel scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

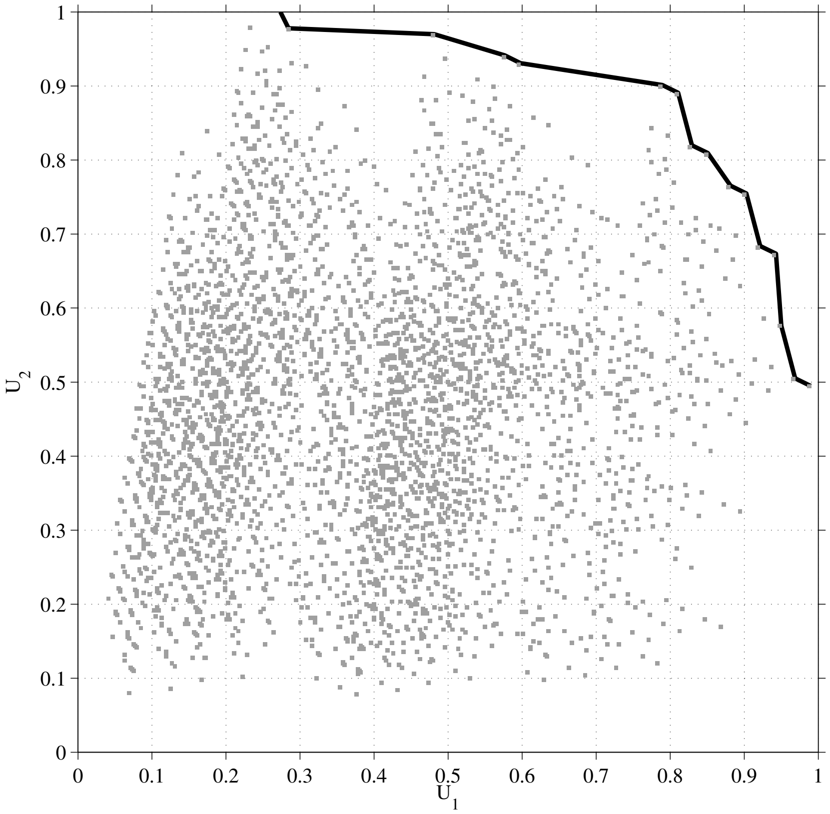


FIGURE A.4: Camera (2012) scenario outcome space.

A.5 Energy (2012 small) Scenario

Domain	Profile 1	Profile 2
Issue 1: 0000-0400	$w_{1,1} = 0.264$	$w_{2,1} = 0.046$
$V_{1,1} = 0$ kW	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0$
$V_{1,2} = 25$ kW	$U_{1,1}(V_{1,2}) = 0.814$	$U_{2,1}(V_{1,2}) = 0.014$
$V_{1,3} = 50$ kW	$U_{1,1}(V_{1,3}) = 0.777$	$U_{2,1}(V_{1,3}) = 0.277$
$V_{1,4} = 75$ kW	$U_{1,1}(V_{1,4}) = 0.185$	$U_{2,1}(V_{1,4}) = 0.883$
$V_{1,5} = 100$ kW	$U_{1,1}(V_{1,5}) = 0$	$U_{2,1}(V_{1,5}) = 1$
Issue 2: 0400-0800	$w_{1,2} = 0.163$	$w_{2,2} = 0.253$
$V_{2,1} = 0$ kW	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0$
$V_{2,2} = 25$ kW	$U_{1,2}(V_{2,2}) = 0.959$	$U_{2,2}(V_{2,2}) = 0.019$
$V_{2,3} = 50$ kW	$U_{1,2}(V_{2,3}) = 0.947$	$U_{2,2}(V_{2,3}) = 0.344$
$V_{2,4} = 75$ kW	$U_{1,2}(V_{2,4}) = 0.084$	$U_{2,2}(V_{2,4}) = 0.895$
$V_{2,5} = 100$ kW	$U_{1,2}(V_{2,5}) = 0$	$U_{2,2}(V_{2,5}) = 1$
Issue 3: 0800-1200	$w_{1,3} = 0.176$	$w_{2,3} = 0.273$
$V_{3,1} = 0$ kW	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0$
$V_{3,2} = 25$ kW	$U_{1,3}(V_{3,2}) = 0.839$	$U_{2,3}(V_{3,2}) = 0.243$
$V_{3,3} = 50$ kW	$U_{1,3}(V_{3,3}) = 0.55$	$U_{2,3}(V_{3,3}) = 0.413$
$V_{3,4} = 75$ kW	$U_{1,3}(V_{3,4}) = 0.482$	$U_{2,3}(V_{3,4}) = 0.713$
$V_{3,5} = 100$ kW	$U_{1,3}(V_{3,5}) = 0$	$U_{2,3}(V_{3,5}) = 1$
Issue 4: 1200-1600	$w_{1,4} = 0.051$	$w_{2,4} = 0.215$
$V_{4,1} = 0$ kW	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0$
$V_{4,2} = 25$ kW	$U_{1,4}(V_{4,2}) = 0.905$	$U_{2,4}(V_{4,2}) = 0.354$
$V_{4,3} = 50$ kW	$U_{1,4}(V_{4,3}) = 0.386$	$U_{2,4}(V_{4,3}) = 0.51$
$V_{4,4} = 75$ kW	$U_{1,4}(V_{4,4}) = 0.265$	$U_{2,4}(V_{4,4}) = 0.555$
$V_{4,5} = 100$ kW	$U_{1,4}(V_{4,5}) = 0$	$U_{2,4}(V_{4,5}) = 1$
Issue 5: 1600-2000	$w_{1,5} = 0.322$	$w_{2,5} = 0.155$
$V_{5,1} = 0$ kW	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0$
$V_{5,2} = 25$ kW	$U_{1,5}(V_{5,2}) = 0.659$	$U_{2,5}(V_{5,2}) = 0.022$
$V_{5,3} = 50$ kW	$U_{1,5}(V_{5,3}) = 0.394$	$U_{2,5}(V_{5,3}) = 0.107$
$V_{5,4} = 75$ kW	$U_{1,5}(V_{5,4}) = 0.192$	$U_{2,5}(V_{5,4}) = 0.749$
$V_{5,5} = 100$ kW	$U_{1,5}(V_{5,5}) = 0$	$U_{2,5}(V_{5,5}) = 1$
Issue 6: 2000-0000	$w_{1,6} = 0.025$	$w_{2,6} = 0.059$
$V_{6,1} = 0$ kW	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 0$
$V_{6,2} = 25$ kW	$U_{1,6}(V_{6,2}) = 0.714$	$U_{2,6}(V_{6,2}) = 0.671$
$V_{6,3} = 50$ kW	$U_{1,6}(V_{6,3}) = 0.675$	$U_{2,6}(V_{6,3}) = 0.741$
$V_{6,4} = 75$ kW	$U_{1,6}(V_{6,4}) = 0.527$	$U_{2,6}(V_{6,4}) = 0.807$
$V_{6,5} = 100$ kW	$U_{1,6}(V_{6,5}) = 0$	$U_{2,6}(V_{6,5}) = 1$

TABLE A.9: Energy (2012 small) scenario specification.

Rank	Agent	Score
1	OMACagent	0.512 ±0.006
2-3	AgentLG	0.486 ±0.004
2-3	TheNegotiator Reloaded	0.480 ±0.005
4-6	IAMhaggler2012	0.451 ±0.006
4-6	CUHKAgent	0.446 ±0.006
6-7	BRAMAgent2	0.425 ±0.011
4-7	Meta-Agent	0.424 ±0.017
8	AgentMR	0.188 ±0.000

TABLE A.10: Scores in the Energy (2012 small) scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

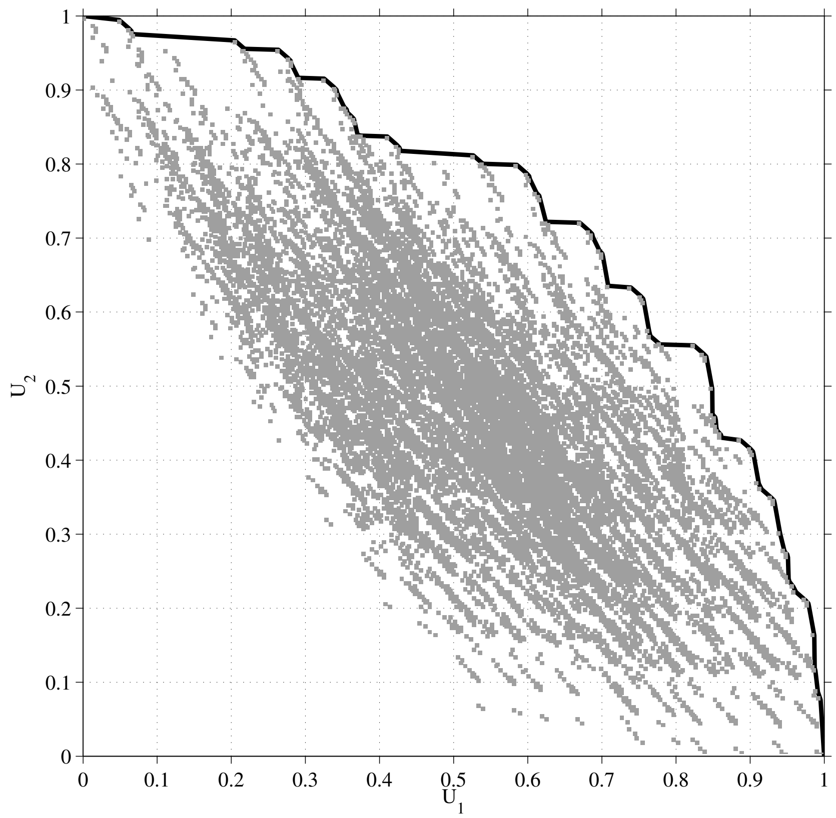


FIGURE A.5: Energy (2012 small) scenario outcome space.

A.6 Energy (2012) Scenario

Domain	Profile 1	Profile 2
Issue 1: 0000-0300	$w_{1,1} = 0.16$	$w_{2,1} = 0.21$
$V_{1,1} = 0$ kW	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0$
$V_{1,2} = 25$ kW	$U_{1,1}(V_{1,2}) = 0.861$	$U_{2,1}(V_{1,2}) = 0.173$
$V_{1,3} = 50$ kW	$U_{1,1}(V_{1,3}) = 0.824$	$U_{2,1}(V_{1,3}) = 0.388$
$V_{1,4} = 75$ kW	$U_{1,1}(V_{1,4}) = 0.044$	$U_{2,1}(V_{1,4}) = 0.63$
$V_{1,5} = 100$ kW	$U_{1,1}(V_{1,5}) = 0$	$U_{2,1}(V_{1,5}) = 1$
Issue 2: 0300-0600	$w_{1,2} = 0.154$	$w_{2,2} = 0.144$
$V_{2,1} = 0$ kW	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0$
$V_{2,2} = 25$ kW	$U_{1,2}(V_{2,2}) = 0.443$	$U_{2,2}(V_{2,2}) = 0.064$
$V_{2,3} = 50$ kW	$U_{1,2}(V_{2,3}) = 0.414$	$U_{2,2}(V_{2,3}) = 0.291$
$V_{2,4} = 75$ kW	$U_{1,2}(V_{2,4}) = 0.248$	$U_{2,2}(V_{2,4}) = 0.592$
$V_{2,5} = 100$ kW	$U_{1,2}(V_{2,5}) = 0$	$U_{2,2}(V_{2,5}) = 1$
Issue 3: 0600-0900	$w_{1,3} = 0.097$	$w_{2,3} = 0.054$
$V_{3,1} = 0$ kW	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0$
$V_{3,2} = 25$ kW	$U_{1,3}(V_{3,2}) = 0.854$	$U_{2,3}(V_{3,2}) = 0.262$
$V_{3,3} = 50$ kW	$U_{1,3}(V_{3,3}) = 0.604$	$U_{2,3}(V_{3,3}) = 0.277$
$V_{3,4} = 75$ kW	$U_{1,3}(V_{3,4}) = 0.41$	$U_{2,3}(V_{3,4}) = 0.822$
$V_{3,5} = 100$ kW	$U_{1,3}(V_{3,5}) = 0$	$U_{2,3}(V_{3,5}) = 1$
Issue 4: 0900-1200	$w_{1,4} = 0.114$	$w_{2,4} = 0.113$
$V_{4,1} = 0$ kW	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0$
$V_{4,2} = 25$ kW	$U_{1,4}(V_{4,2}) = 0.749$	$U_{2,4}(V_{4,2}) = 0.461$
$V_{4,3} = 50$ kW	$U_{1,4}(V_{4,3}) = 0.447$	$U_{2,4}(V_{4,3}) = 0.476$
$V_{4,4} = 75$ kW	$U_{1,4}(V_{4,4}) = 0.056$	$U_{2,4}(V_{4,4}) = 0.537$
$V_{4,5} = 100$ kW	$U_{1,4}(V_{4,5}) = 0$	$U_{2,4}(V_{4,5}) = 1$
Issue 5: 1200-1500	$w_{1,5} = 0.104$	$w_{2,5} = 0.111$
$V_{5,1} = 0$ kW	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0$
$V_{5,2} = 25$ kW	$U_{1,5}(V_{5,2}) = 0.914$	$U_{2,5}(V_{5,2}) = 0.34$
$V_{5,3} = 50$ kW	$U_{1,5}(V_{5,3}) = 0.66$	$U_{2,5}(V_{5,3}) = 0.359$
$V_{5,4} = 75$ kW	$U_{1,5}(V_{5,4}) = 0.285$	$U_{2,5}(V_{5,4}) = 0.461$
$V_{5,5} = 100$ kW	$U_{1,5}(V_{5,5}) = 0$	$U_{2,5}(V_{5,5}) = 1$
Issue 6: 1500-1800	$w_{1,6} = 0.068$	$w_{2,6} = 0.139$
$V_{6,1} = 0$ kW	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 0$
$V_{6,2} = 25$ kW	$U_{1,6}(V_{6,2}) = 0.67$	$U_{2,6}(V_{6,2}) = 0.392$
$V_{6,3} = 50$ kW	$U_{1,6}(V_{6,3}) = 0.507$	$U_{2,6}(V_{6,3}) = 0.797$
$V_{6,4} = 75$ kW	$U_{1,6}(V_{6,4}) = 0.147$	$U_{2,6}(V_{6,4}) = 0.93$
$V_{6,5} = 100$ kW	$U_{1,6}(V_{6,5}) = 0$	$U_{2,6}(V_{6,5}) = 1$
Issue 7: 1800-2100	$w_{1,7} = 0.126$	$w_{2,7} = 0.188$
$V_{7,1} = 0$ kW	$U_{1,7}(V_{7,1}) = 1$	$U_{2,7}(V_{7,1}) = 0$
$V_{7,2} = 25$ kW	$U_{1,7}(V_{7,2}) = 0.505$	$U_{2,7}(V_{7,2}) = 0.16$
$V_{7,3} = 50$ kW	$U_{1,7}(V_{7,3}) = 0.223$	$U_{2,7}(V_{7,3}) = 0.527$
$V_{7,4} = 75$ kW	$U_{1,7}(V_{7,4}) = 0.188$	$U_{2,7}(V_{7,4}) = 0.972$
$V_{7,5} = 100$ kW	$U_{1,7}(V_{7,5}) = 0$	$U_{2,7}(V_{7,5}) = 1$
Issue 8: 2100-0000	$w_{1,8} = 0.176$	$w_{2,8} = 0.041$
$V_{8,1} = 0$ kW	$U_{1,8}(V_{8,1}) = 1$	$U_{2,8}(V_{8,1}) = 0$
$V_{8,2} = 25$ kW	$U_{1,8}(V_{8,2}) = 0.942$	$U_{2,8}(V_{8,2}) = 0.232$
$V_{8,3} = 50$ kW	$U_{1,8}(V_{8,3}) = 0.471$	$U_{2,8}(V_{8,3}) = 0.289$
$V_{8,4} = 75$ kW	$U_{1,8}(V_{8,4}) = 0.33$	$U_{2,8}(V_{8,4}) = 0.846$
$V_{8,5} = 100$ kW	$U_{1,8}(V_{8,5}) = 0$	$U_{2,8}(V_{8,5}) = 1$

TABLE A.11: Energy (2012) scenario specification.

Rank	Agent	Score
1	OMACagent	0.457 ±0.003
2	AgentLG	0.435 ±0.008
3-4	BRAMAgent2	0.411 ±0.005
3-4	IAMhaggler2012	0.409 ±0.004
5-6	CUHKAgent	0.389 ±0.006
5-7	TheNegotiator Reloaded	0.378 ±0.013
6-7	Meta-Agent	0.361 ±0.002
8	AgentMR	0.221 ±0.002

Table A.12: Scores in the Energy (2012) scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

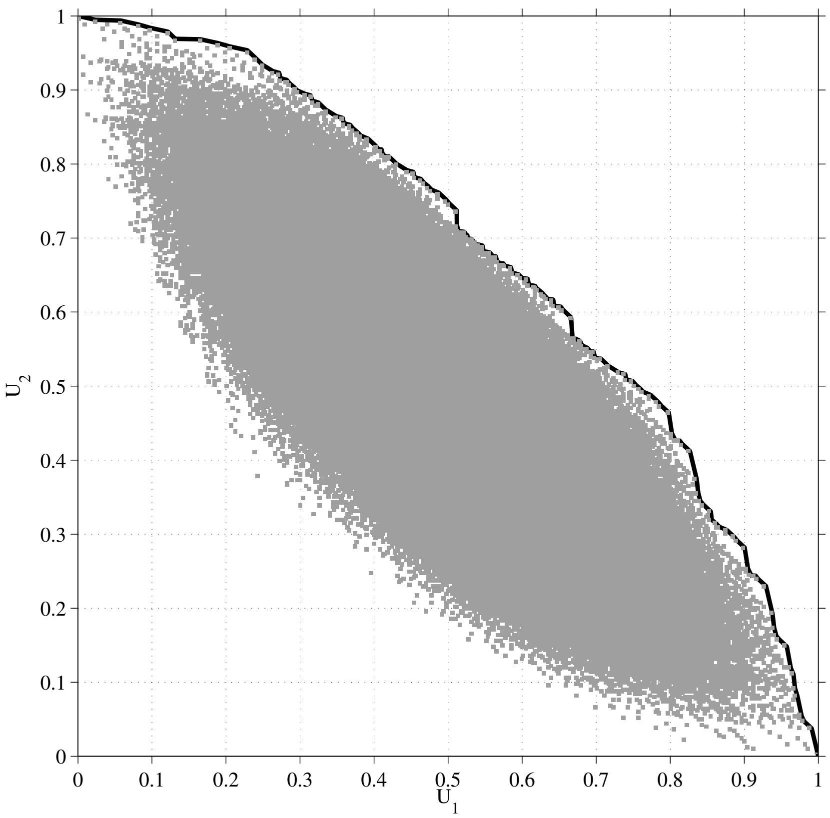


Figure A.6: Energy (2012) scenario outcome space.

A.7 England vs Zimbabwe (2012) Scenario

Domain	Profile 1	Profile 2
Issue 1: Size of Fund	$w_{1,1} = 0.303$	$w_{2,1} = 0.197$
$V_{1,1}$ = $100 Billion	$U_{1,1}(V_{1,1}) = 0.556$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2}$ = $50 Billion	$U_{1,1}(V_{1,2}) = 0.778$	$U_{2,1}(V_{1,2}) = 0.778$
$V_{1,3}$ = $10 billion	$U_{1,1}(V_{1,3}) = 1$	$U_{2,1}(V_{1,3}) = 0.556$
$V_{1,4}$ = No agreement	$U_{1,1}(V_{1,4}) = 0.111$	$U_{2,1}(V_{1,4}) = 0.111$
Issue 2: Impact on Other Aid	$w_{1,2} = 0.303$	$w_{2,2} = 0.201$
$V_{2,1}$ = No reduction	$U_{1,2}(V_{2,1}) = 0.375$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = Reduction equal to half of fund size	$U_{1,2}(V_{2,2}) = 0.75$	$U_{2,2}(V_{2,2}) = 0.625$
$V_{2,3}$ = Reduction equal to fund size	$U_{1,2}(V_{2,3}) = 1$	$U_{2,2}(V_{2,3}) = 0.375$
$V_{2,4}$ = No agreement	$U_{1,2}(V_{2,4}) = 0.125$	$U_{2,2}(V_{2,4}) = 0.125$
Issue 3: Zimbabwe Trade Policy	$w_{1,3} = 0.049$	$w_{2,3} = 0.154$
$V_{3,1}$ = Zimbabwe will reduce tariffs on imports	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0.111$
$V_{3,2}$ = Zimbabwe will increase tariffs on imports	$U_{1,3}(V_{3,2}) = 0.083$	$U_{2,3}(V_{3,2}) = 1$
$V_{3,3}$ = No agreement	$U_{1,3}(V_{3,3}) = 0.583$	$U_{2,3}(V_{3,3}) = 0.556$
Issue 4: England Trade Policy	$w_{1,4} = 0.049$	$w_{2,4} = 0.154$
$V_{4,1}$ = England will reduce imports	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0.053$
$V_{4,2}$ = England will increase imports	$U_{1,4}(V_{4,2}) = 0.1$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3}$ = No agreement	$U_{1,4}(V_{4,3}) = 0.6$	$U_{2,4}(V_{4,3}) = 0.474$
Issue 5: Forum on Other Health Issues	$w_{1,5} = 0.295$	$w_{2,5} = 0.293$
$V_{5,1}$ = Creation of fund	$U_{1,5}(V_{5,1}) = 0.7$	$U_{2,5}(V_{5,1}) = 1$
$V_{5,2}$ = Creation of committee to discuss creation of fund	$U_{1,5}(V_{5,2}) = 1$	$U_{2,5}(V_{5,2}) = 0.818$
$V_{5,3}$ = Creation of committee to develop agenda	$U_{1,5}(V_{5,3}) = 0.4$	$U_{2,5}(V_{5,3}) = 0.636$
$V_{5,4}$ = No	$U_{1,5}(V_{5,4}) = 0.1$	$U_{2,5}(V_{5,4}) = 0.091$

Table A.13: England vs Zimbabwe (2012) scenario specification.

Rank	Agent	Score
1-2	AgentLG	0.664 ±0.004
1-4	CUHKAgent	0.650 ±0.008
2-4	TheNegotiator Reloaded	0.648 ±0.005
2-4	OMACagent	0.647 ±0.001
5-6	IAMhaggler2012	0.624 ±0.005
5-6	Meta-Agent	0.616 ±0.006
7	BRAMAgent2	0.600 ±0.005
8	AgentMR	0.410 ±0.006

Table A.14: Scores in the England vs Zimbabwe (2012) scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

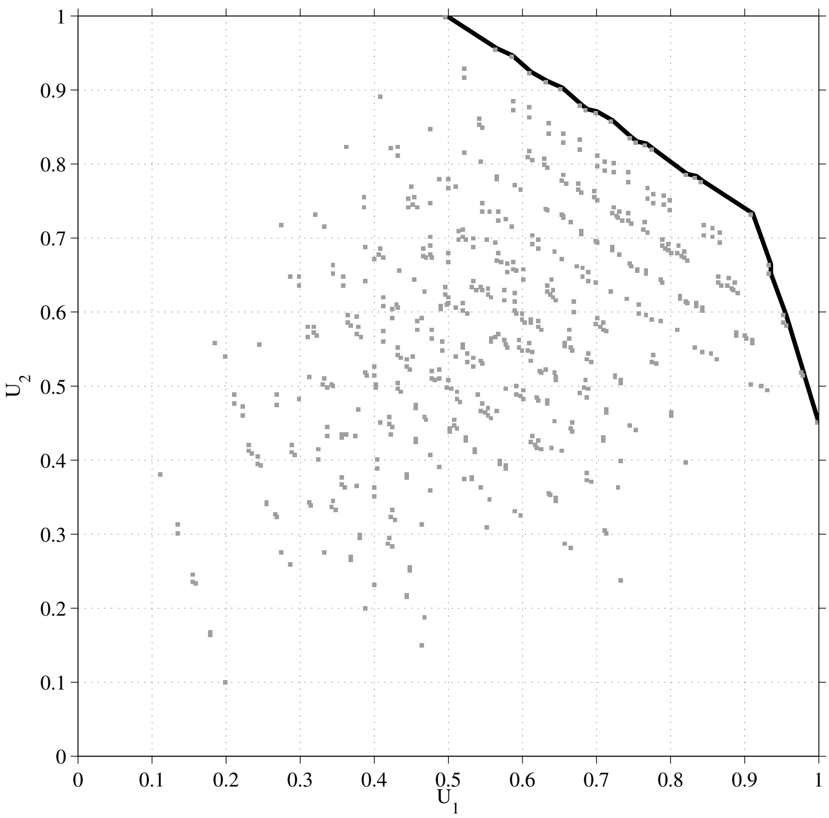


Figure A.7: England vs Zimbabwe (2012) scenario outcome space.

A.8 Fifty fifty Scenario

Domain	Profile 1	Profile 2
Issue 1: Fifty_Fifty	$w_{1,1} = 1$	$w_{2,1} = 1$
$V_{1,1} = 100\_0$	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0$
$V_{1,2} = 90\_10$	$U_{1,1}(V_{1,2}) = 0.9$	$U_{2,1}(V_{1,2}) = 0.1$
$V_{1,3} = 80\_20$	$U_{1,1}(V_{1,3}) = 0.8$	$U_{2,1}(V_{1,3}) = 0.2$
$V_{1,4} = 70\_30$	$U_{1,1}(V_{1,4}) = 0.7$	$U_{2,1}(V_{1,4}) = 0.3$
$V_{1,5} = 60\_40$	$U_{1,1}(V_{1,5}) = 0.6$	$U_{2,1}(V_{1,5}) = 0.4$
$V_{1,6} = 50\_50$	$U_{1,1}(V_{1,6}) = 0.5$	$U_{2,1}(V_{1,6}) = 0.5$
$V_{1,7} = 40\_60$	$U_{1,1}(V_{1,7}) = 0.4$	$U_{2,1}(V_{1,7}) = 0.6$
$V_{1,8} = 30\_70$	$U_{1,1}(V_{1,8}) = 0.3$	$U_{2,1}(V_{1,8}) = 0.7$
$V_{1,9} = 20\_80$	$U_{1,1}(V_{1,9}) = 0.2$	$U_{2,1}(V_{1,9}) = 0.8$
$V_{1,10} = 10\_90$	$U_{1,1}(V_{1,10}) = 0.1$	$U_{2,1}(V_{1,10}) = 0.9$
$V_{1,11} = 0\_100$	$U_{1,1}(V_{1,11}) = 0$	$U_{2,1}(V_{1,11}) = 1$

TABLE A.15: Fifty fifty scenario specification.

Rank	Agent	Score
1-3	IAMhaggler2012	0.376 ±0.018
1-2	CUHKAgent	0.371 ±0.005
2-3	OMACagent	0.361 ±0.006
4-6	AgentLG	0.343 ±0.012
4-6	BRAMAgent2	0.341 ±0.008
4-7	TheNegotiator Reloaded	0.329 ±0.008
6-7	Meta-Agent	0.318 ±0.013
8	AgentMR	0.235 ±0.007

TABLE A.16: Scores in the Fifty fifty scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

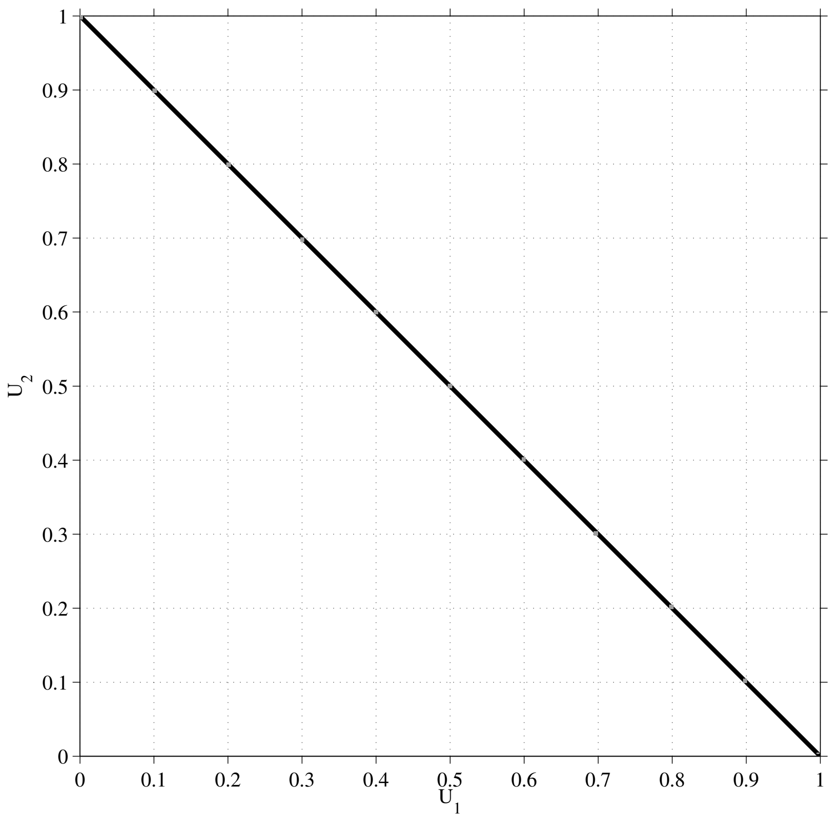


FIGURE A.8: Fifty fifty scenario outcome space.

A.9 Fitness Scenario

Domain	Profile 1	Profile 2
Issue 1: kind of fitness	$w_{1,1} = 0.154$	$w_{2,1} = 0.304$
$V_{1,1} =$ SWIMMING	$U_{1,1}(V_{1,1}) = 0.6$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2} =$ YOGA	$U_{1,1}(V_{1,2}) = 0.8$	$U_{2,1}(V_{1,2}) = 0.6$
$V_{1,3} =$ AEROBICS	$U_{1,1}(V_{1,3}) = 0.4$	$U_{2,1}(V_{1,3}) = 0.4$
$V_{1,4} =$ RUNNING	$U_{1,1}(V_{1,4}) = 0.2$	$U_{2,1}(V_{1,4}) = 0.2$
$V_{1,5} =$ TENNIS	$U_{1,1}(V_{1,5}) = 1$	$U_{2,1}(V_{1,5}) = 0.8$
Issue 2: time to do	$w_{1,2} = 0.045$	$w_{2,2} = 0.098$
$V_{2,1} =$ 30MINUTES	$U_{1,2}(V_{2,1}) = 0.75$	$U_{2,2}(V_{2,1}) = 0.25$
$V_{2,2} =$ 1HOUR	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.75$
$V_{2,3} =$ 3HOUR	$U_{1,2}(V_{2,3}) = 0.5$	$U_{2,2}(V_{2,3}) = 1$
$V_{2,4} =$ 5HOUR	$U_{1,2}(V_{2,4}) = 0.25$	$U_{2,2}(V_{2,4}) = 0.5$
Issue 3: distance	$w_{1,3} = 0.298$	$w_{2,3} = 0.201$
$V_{3,1} =$ 0KM	$U_{1,3}(V_{3,1}) = 0.75$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2} =$ 1KM	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.75$
$V_{3,3} =$ 40KM	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.5$
$V_{3,4} =$ 80KM	$U_{1,3}(V_{3,4}) = 0.25$	$U_{2,3}(V_{3,4}) = 0.25$
Issue 4: intensity	$w_{1,4} = 0.299$	$w_{2,4} = 0.098$
$V_{4,1} =$ LIGHT	$U_{1,4}(V_{4,1}) = 0.5$	$U_{2,4}(V_{4,1}) = 1$
$V_{4,2} =$ MODERATE	$U_{1,4}(V_{4,2}) = 0.75$	$U_{2,4}(V_{4,2}) = 0.75$
$V_{4,3} =$ AS TRAINING	$U_{1,4}(V_{4,3}) = 1$	$U_{2,4}(V_{4,3}) = 0.5$
$V_{4,4} =$ AS BOOT-CAMP	$U_{1,4}(V_{4,4}) = 0.25$	$U_{2,4}(V_{4,4}) = 0.25$
Issue 5: Price($)	$w_{1,5} = 0.204$	$w_{2,5} = 0.299$
$V_{5,1} = 0$	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0$
$V_{5,2} = 1$	$U_{1,5}(V_{5,2}) = 0.9$	$U_{2,5}(V_{5,2}) = 0.1$
$V_{5,3} = 2$	$U_{1,5}(V_{5,3}) = 0.8$	$U_{2,5}(V_{5,3}) = 0.2$
$V_{5,4} = 3$	$U_{1,5}(V_{5,4}) = 0.7$	$U_{2,5}(V_{5,4}) = 0.3$
$V_{5,5} = 4$	$U_{1,5}(V_{5,5}) = 0.6$	$U_{2,5}(V_{5,5}) = 0.4$
$V_{5,6} = 5$	$U_{1,5}(V_{5,6}) = 0.5$	$U_{2,5}(V_{5,6}) = 0.5$
$V_{5,7} = 6$	$U_{1,5}(V_{5,7}) = 0.4$	$U_{2,5}(V_{5,7}) = 0.6$
$V_{5,8} = 7$	$U_{1,5}(V_{5,8}) = 0.3$	$U_{2,5}(V_{5,8}) = 0.7$
$V_{5,9} = 8$	$U_{1,5}(V_{5,9}) = 0.2$	$U_{2,5}(V_{5,9}) = 0.8$
$V_{5,10} = 9$	$U_{1,5}(V_{5,10}) = 0.1$	$U_{2,5}(V_{5,10}) = 0.9$
$V_{5,11} = 10$	$U_{1,5}(V_{5,11}) = 0$	$U_{2,5}(V_{5,11}) = 1$

TABLE A.17: Fitness scenario specification.

Rank	Agent	Score
1	AgentLG	0.654 ±0.006
2-3	TheNegotiator Reloaded	0.623 ±0.004
2-4	CUHKAgent	0.620 ±0.006
3-5	OMACagent	0.611 ±0.002
4-6	IAMhaggler2012	0.591 ±0.015
5-6	Meta-Agent	0.578 ±0.008
7	BRAMAgent2	0.562 ±0.010
8	AgentMR	0.284 ±0.009

TABLE A.18: Scores in the Fitness scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

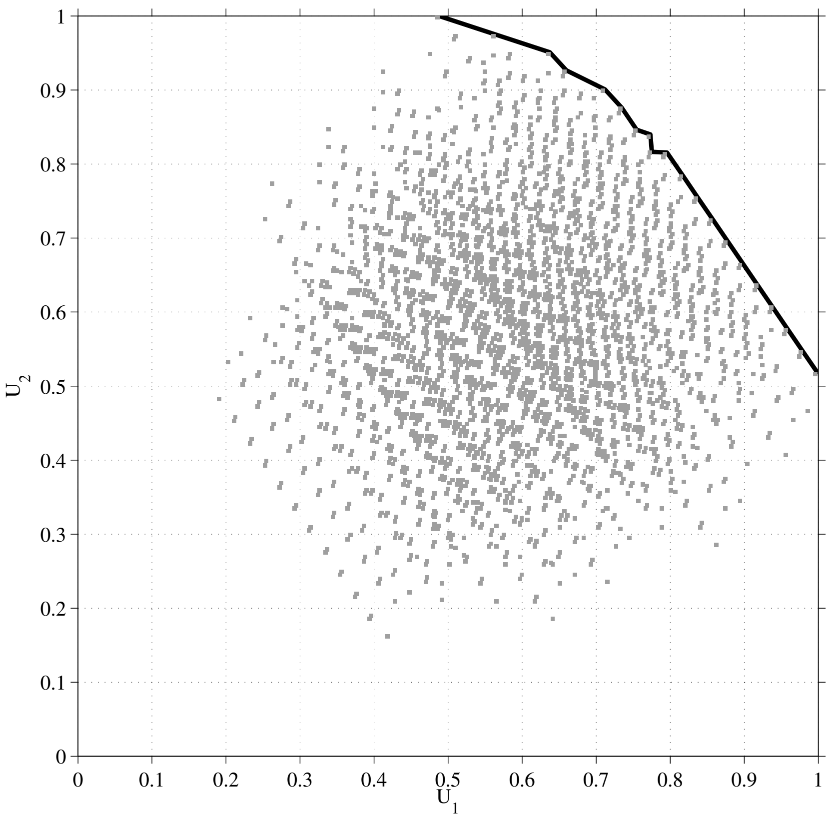


FIGURE A.9: Fitness scenario outcome space.

A.10 Flight Booking Scenario

Domain	Profile 1	Profile 2
Issue 1: price	$w_{1,1} = 0.35$	$w_{2,1} = 0.7$
$V_{1,1} = 150000$YEN	$U_{1,1}(V_{1,1}) = 0.25$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2} = 200000$YEN	$U_{1,1}(V_{1,2}) = 0.5$	$U_{2,1}(V_{1,2}) = 0.75$
$V_{1,3} = 250000$YEN	$U_{1,1}(V_{1,3}) = 0.75$	$U_{2,1}(V_{1,3}) = 0.5$
$V_{1,4} = 300000$YEN	$U_{1,1}(V_{1,4}) = 1$	$U_{2,1}(V_{1,4}) = 0.25$
Issue 2: DepartureAirPort	$w_{1,2} = 0.35$	$w_{2,2} = 0.15$
$V_{2,1} =$ CENTRAIR	$U_{1,2}(V_{2,1}) = 0.667$	$U_{2,2}(V_{2,1}) = 0.333$
$V_{2,2} =$ NARITA	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.667$
$V_{2,3} =$ KANSAI	$U_{1,2}(V_{2,3}) = 0.333$	$U_{2,2}(V_{2,3}) = 1$
Issue 3: DepartureDate	$w_{1,3} = 0.3$	$w_{2,3} = 0.15$
$V_{3,1} =$ SEP2_2011	$U_{1,3}(V_{3,1}) = 0.333$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2} =$ SEP3_2011	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.667$
$V_{3,3} =$ SEP4_2011	$U_{1,3}(V_{3,3}) = 0.667$	$U_{2,3}(V_{3,3}) = 0.333$

TABLE A.19: Flight Booking scenario specification.

Rank	Agent	Score
1-4	CUHKAgent	0.625 ±0.006
1-3	TheNegotiator Reloaded	0.622 ±0.001
1-4	BRAMAgent2	0.620 ±0.003
2-4	OMACagent	0.617 ±0.003
5	Meta-Agent	0.605 ±0.000
6	IAMhaggler2012	0.599 ±0.000
7	AgentLG	0.581 ±0.003
8	AgentMR	0.412 ±0.001

Table A.20: Scores in the Flight Booking scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

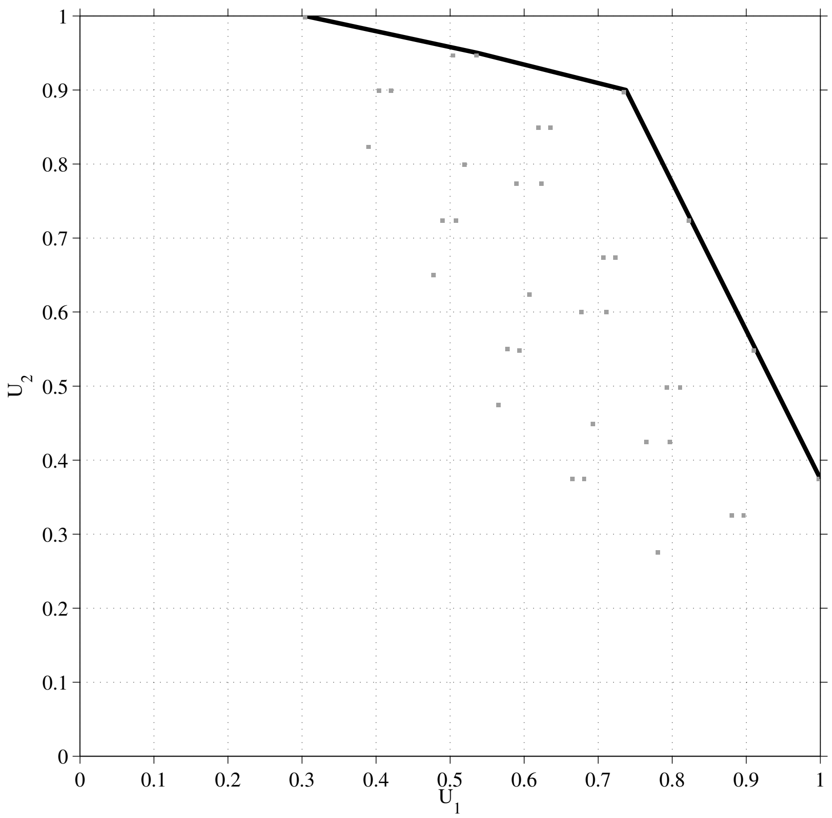


Figure A.10: Flight Booking scenario outcome space.

A.11 Housekeeping Scenario

Domain	Profile 1	Profile 2
Issue 1: Floor Mopping	$w_{1,1} = 0.3$	$w_{2,1} = 0.1$
$V_{1,1}$ = Husband	$U_{1,1}(V_{1,1}) = 0.1$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2}$ = Wife	$U_{1,1}(V_{1,2}) = 1$	$U_{2,1}(V_{1,2}) = 0.5$
$V_{1,3}$ = Both	$U_{1,1}(V_{1,3}) = 0.3$	$U_{2,1}(V_{1,3}) = 0.8$
$V_{1,4}$ = Maid	$U_{1,1}(V_{1,4}) = 0.7$	$U_{2,1}(V_{1,4}) = 0.3$
Issue 2: Dishes Cleaning	$w_{1,2} = 0.1$	$w_{2,2} = 0.4$
$V_{2,1}$ = Husband	$U_{1,2}(V_{2,1}) = 0.4$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = Wife	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.1$
$V_{2,3}$ = Both	$U_{1,2}(V_{2,3}) = 0.85$	$U_{2,2}(V_{2,3}) = 0.3$
Issue 3: Laundry	$w_{1,3} = 0.2$	$w_{2,3} = 0.2$
$V_{3,1}$ = Husband	$U_{1,3}(V_{3,1}) = 0.5$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = Wife	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.6$
Issue 4: Cooking	$w_{1,4} = 0.2$	$w_{2,4} = 0.15$
$V_{4,1}$ = Husband	$U_{1,4}(V_{4,1}) = 0.1$	$U_{2,4}(V_{4,1}) = 0.8$
$V_{4,2}$ = Wife	$U_{1,4}(V_{4,2}) = 0.7$	$U_{2,4}(V_{4,2}) = 0.7$
$V_{4,3}$ = Both	$U_{1,4}(V_{4,3}) = 0.4$	$U_{2,4}(V_{4,3}) = 1$
$V_{4,4}$ = Take-away food	$U_{1,4}(V_{4,4}) = 1$	$U_{2,4}(V_{4,4}) = 0.4$
Issue 5: Gardening	$w_{1,5} = 0.2$	$w_{2,5} = 0.15$
$V_{5,1}$ = Husband	$U_{1,5}(V_{5,1}) = 0.5$	$U_{2,5}(V_{5,1}) = 1$
$V_{5,2}$ = Wife	$U_{1,5}(V_{5,2}) = 0.7$	$U_{2,5}(V_{5,2}) = 0.1$
$V_{5,3}$ = Both	$U_{1,5}(V_{5,3}) = 1$	$U_{2,5}(V_{5,3}) = 0.4$
$V_{5,4}$ = Gardener	$U_{1,5}(V_{5,4}) = 0.1$	$U_{2,5}(V_{5,4}) = 0.9$

Table A.21: Housekeeping scenario specification.

Rank	Agent	Score
1-3	AgentLG	0.615 ±0.008
1-3	TheNegotiator Reloaded	0.613 ±0.005
1-4	OMACagent	0.613 ±0.007
3-5	CUHKAgent	0.603 ±0.006
4-5	BRAMAgent2	0.595 ±0.004
6	IAMhaggler2012	0.586 ±0.004
7	Meta-Agent	0.576 ±0.005
8	AgentMR	0.201 ±0.000

TABLE A.22: Scores in the Housekeeping scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

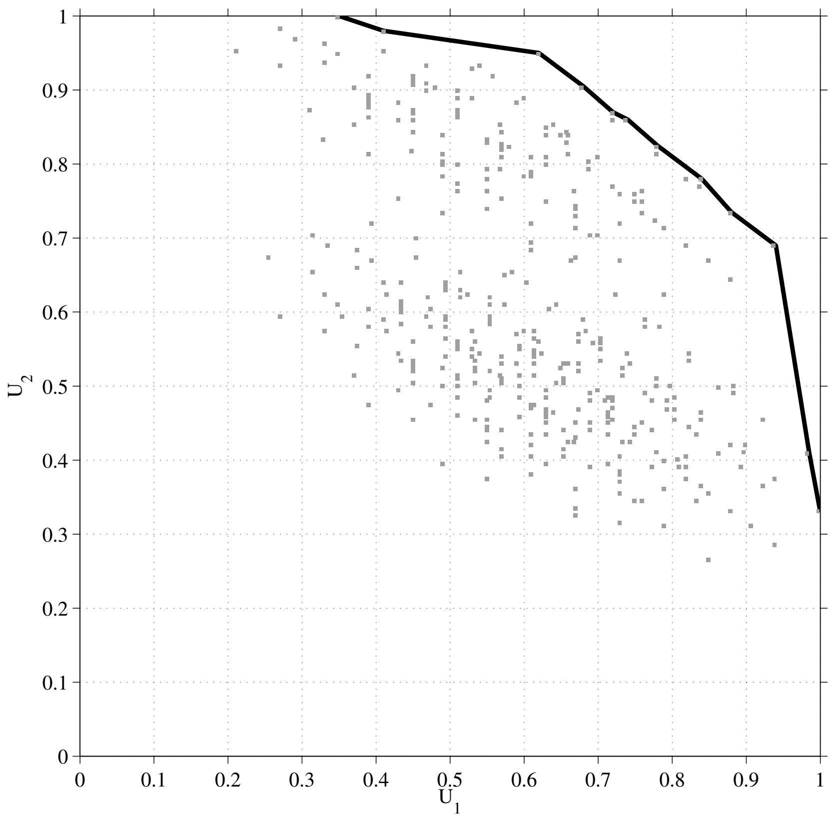


FIGURE A.11: Housekeeping scenario outcome space.

A.12 IS BT Acquisition (2012) Scenario

Domain	Profile 1	Profile 2
Issue 1: Price	$w_{1,1} = 0.3$	$w_{2,1} = 0.5$
$V_{1,1}$ = 1 MILLION \$	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.8$
$V_{1,2}$ = 2.5 MILLION \$	$U_{1,1}(V_{1,2}) = 0.9$	$U_{2,1}(V_{1,2}) = 0.8$
$V_{1,3}$ = 5 MILLION \$	$U_{1,1}(V_{1,3}) = 0.8$	$U_{2,1}(V_{1,3}) = 0.9$
$V_{1,4}$ = 8 MILLION \$	$U_{1,1}(V_{1,4}) = 0.8$	$U_{2,1}(V_{1,4}) = 1$
Issue 2: IP	$w_{1,2} = 0.3$	$w_{2,2} = 0.04$
$V_{2,1}$ = IS RECEIVES ALL OF THE IP	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0.6$
$V_{2,2}$ = IS RECEIVES MOST OF THE IP	$U_{1,2}(V_{2,2}) = 0.4$	$U_{2,2}(V_{2,2}) = 0.7$
$V_{2,3}$ = BI-TECH FOUNDERS MAINTAIN ALL IP	$U_{1,2}(V_{2,3}) = 0.9$	$U_{2,2}(V_{2,3}) = 1$
Issue 3: Stocks	$w_{1,3} = 0.15$	$w_{2,3} = 0.2$
$V_{3,1}$ = BI-TECH FOUNDERS GET 2%	$U_{1,3}(V_{3,1}) = 0.5$	$U_{2,3}(V_{3,1}) = 0.5$
$V_{3,2}$ = BI-TECH FOUNDERS GET 2% + JOBS AT IS	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.9$
$V_{3,3}$ = BI-TECH FOUNDERS GET 5%	$U_{1,3}(V_{3,3}) = 0.4$	$U_{2,3}(V_{3,3}) = 0.6$
$V_{3,4}$ = BI-TECH FOUNDERS GET 5% + JOBS	$U_{1,3}(V_{3,4}) = 0.7$	$U_{2,3}(V_{3,4}) = 1$
Issue 4: EmployeeAgreements	$w_{1,4} = 0.05$	$w_{2,4} = 0.06$
$V_{4,1}$ = SALARY RAISE OF 15%	$U_{1,4}(V_{4,1}) = 0.857$	$U_{2,4}(V_{4,1}) = 1$
$V_{4,2}$ = SAME CONDITIONS	$U_{1,4}(V_{4,2}) = 0.857$	$U_{2,4}(V_{4,2}) = 0.778$
$V_{4,3}$ = PRIVATE CONTRACTS	$U_{1,4}(V_{4,3}) = 1$	$U_{2,4}(V_{4,3}) = 0.889$
$V_{4,4}$ = HALF FIRED AND HALF PRIVATE CONTRACTS	$U_{1,4}(V_{4,4}) = 0.429$	$U_{2,4}(V_{4,4}) = 0.444$
Issue 5: Legal Liability	$w_{1,5} = 0.2$	$w_{2,5} = 0.2$
$V_{5,1}$ = PAST ACTIVITIES REMAINS WITH BI-TECH	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0.7$
$V_{5,2}$ = IS LIABLE FOR ALL ACTIVITIES	$U_{1,5}(V_{5,2}) = 0.7$	$U_{2,5}(V_{5,2}) = 1$

TABLE A.23: IS BT Acquisition (2012) scenario specification.

Rank	Agent	Score
1-2	TheNegotiator Reloaded	0.786 ±0.000
1-3	CUHKAgent	0.776 ±0.011
2-3	AgentLG	0.772 ±0.007
4-6	IAMhaggler2012	0.758 ±0.007
4-7	OMACagent	0.749 ±0.003
4-7	BRAMAgent2	0.744 ±0.016
5-7	Meta-Agent	0.733 ±0.014
8	AgentMR	0.529 ±0.017

TABLE A.24: Scores in the IS BT Acquisition (2012) scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

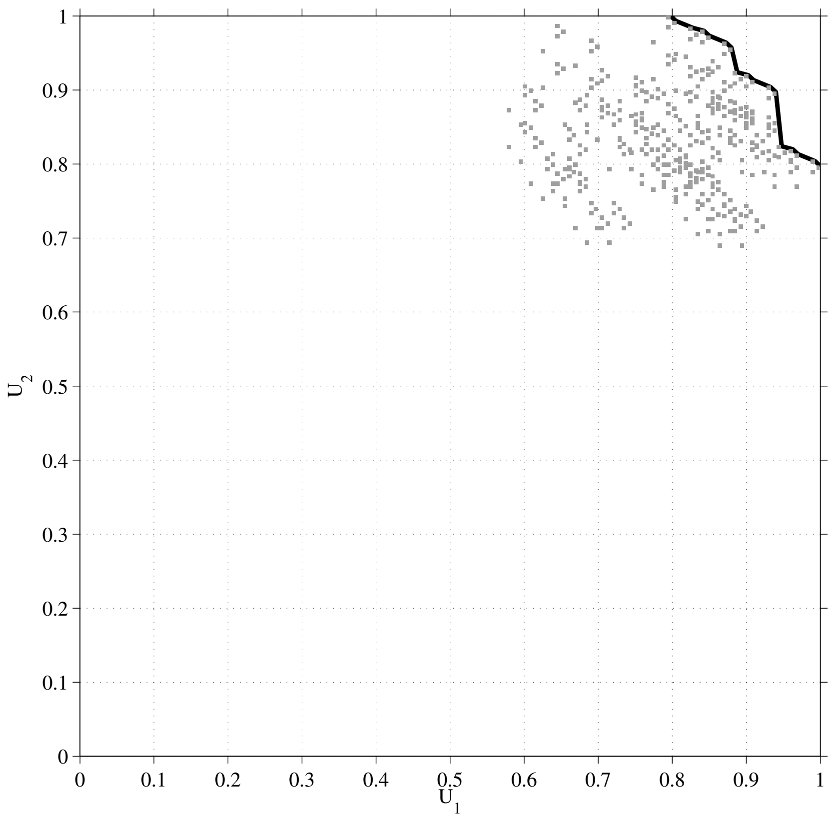


FIGURE A.12: IS BT Acquisition (2012) scenario outcome space.

A.13 Music Collection Scenario

Domain	Profile 1	Profile 2
Issue 1: Classical	$w_{1,1} = 0.055$	$w_{2,1} = 0.299$
$V_{1,1}$ = Bach	$U_{1,1}(V_{1,1}) = 0.25$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2}$ = Mozart	$U_{1,1}(V_{1,2}) = 0.75$	$U_{2,1}(V_{1,2}) = 0.5$
$V_{1,3}$ = Beethoven	$U_{1,1}(V_{1,3}) = 0.5$	$U_{2,1}(V_{1,3}) = 0.25$
$V_{1,4}$ = Prokofiev	$U_{1,1}(V_{1,4}) = 1$	$U_{2,1}(V_{1,4}) = 0.75$
Issue 2: Rock	$w_{1,2} = 0.204$	$w_{2,2} = 0.233$
$V_{2,1}$ = Chuck Berry	$U_{1,2}(V_{2,1}) = 0.4$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = The Beatles	$U_{1,2}(V_{2,2}) = 0.2$	$U_{2,2}(V_{2,2}) = 0.2$
$V_{2,3}$ = The Doors	$U_{1,2}(V_{2,3}) = 0.6$	$U_{2,2}(V_{2,3}) = 0.6$
$V_{2,4}$ = Camel	$U_{1,2}(V_{2,4}) = 1$	$U_{2,2}(V_{2,4}) = 0.8$
$V_{2,5}$ = Nirvana	$U_{1,2}(V_{2,5}) = 0.4$	$U_{2,2}(V_{2,5}) = 0.4$
Issue 3: Jazz	$w_{1,3} = 0.352$	$w_{2,3} = 0.085$
$V_{3,1}$ = Louie Armstrong	$U_{1,3}(V_{3,1}) = 0.833$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = Charlie Parker	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.833$
$V_{3,3}$ = Miles Davis	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.667$
$V_{3,4}$ = John Coltrane	$U_{1,3}(V_{3,4}) = 0.333$	$U_{2,3}(V_{3,4}) = 0.5$
$V_{3,5}$ = Herbie Hancock	$U_{1,3}(V_{3,5}) = 0.167$	$U_{2,3}(V_{3,5}) = 0.167$
$V_{3,6}$ = Cal Tjader	$U_{1,3}(V_{3,6}) = 0.667$	$U_{2,3}(V_{3,6}) = 0.333$
Issue 4: Pop	$w_{1,4} = 0.065$	$w_{2,4} = 0.173$
$V_{4,1}$ = Michael Jackson	$U_{1,4}(V_{4,1}) = 0.333$	$U_{2,4}(V_{4,1}) = 0.333$
$V_{4,2}$ = Madonna	$U_{1,4}(V_{4,2}) = 1$	$U_{2,4}(V_{4,2}) = 0.667$
$V_{4,3}$ = Elton John	$U_{1,4}(V_{4,3}) = 0.667$	$U_{2,4}(V_{4,3}) = 1$
Issue 5: Brazil	$w_{1,5} = 0.124$	$w_{2,5} = 0.109$
$V_{5,1}$ = Gilberto Gil	$U_{1,5}(V_{5,1}) = 0.333$	$U_{2,5}(V_{5,1}) = 0.667$
$V_{5,2}$ = Antonio Carlos Jobim	$U_{1,5}(V_{5,2}) = 1$	$U_{2,5}(V_{5,2}) = 1$
$V_{5,3}$ = Astrud Gilberto	$U_{1,5}(V_{5,3}) = 0.667$	$U_{2,5}(V_{5,3}) = 0.333$
Issue 6: Latin	$w_{1,6} = 0.2$	$w_{2,6} = 0.1$
$V_{6,1}$ = Eddie Palmieri	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 0.25$
$V_{6,2}$ = Maraca	$U_{1,6}(V_{6,2}) = 0.75$	$U_{2,6}(V_{6,2}) = 0.5$
$V_{6,3}$ = Gypsy Kings	$U_{1,6}(V_{6,3}) = 0.25$	$U_{2,6}(V_{6,3}) = 1$
$V_{6,4}$ = Tito Puente	$U_{1,6}(V_{6,4}) = 0.5$	$U_{2,6}(V_{6,4}) = 0.75$

Table A.25: Music Collection scenario specification.

Rank	Agent	Score
1	CUHKAgent	0.767 ±0.004
2-3	AgentLG	0.759 ±0.004
2-3	TheNegotiator Reloaded	0.756 ±0.003
4	OMACagent	0.745 ±0.004
5	IAMhaggler2012	0.735 ±0.002
6	BRAMAgent2	0.725 ±0.002
7	Meta-Agent	0.701 ±0.006
8	AgentMR	0.511 ±0.002

TABLE A.26: Scores in the Music Collection scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

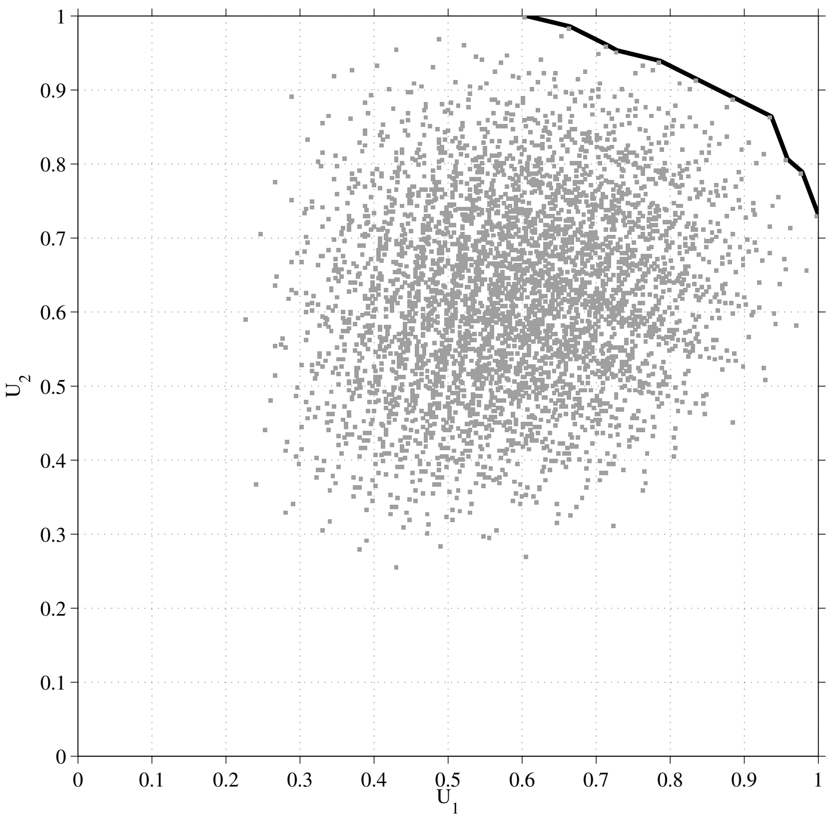


FIGURE A.13: Music Collection scenario outcome space.

A.14 Outfit Scenario

Domain	Profile 1	Profile 2
Issue 1: shirts	$w_{1,1} = 0.147$	$w_{2,1} = 0.379$
$V_{1,1}$ = T-SHIRT	$U_{1,1}(V_{1,1}) = 0.2$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2}$ = BLOUSE	$U_{1,1}(V_{1,2}) = 1$	$U_{2,1}(V_{1,2}) = 0.2$
$V_{1,3}$ = POLO SHIRT	$U_{1,1}(V_{1,3}) = 0.4$	$U_{2,1}(V_{1,3}) = 0.6$
$V_{1,4}$ = SWEATERS	$U_{1,1}(V_{1,4}) = 0.6$	$U_{2,1}(V_{1,4}) = 0.8$
Issue 2: pants	$w_{1,2} = 0.204$	$w_{2,2} = 0.318$
$V_{2,1}$ = DENIM	$U_{1,2}(V_{2,1}) = 0.3$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = LEATHER PANTS	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.1$
$V_{2,3}$ = CLASSIC PANTS	$U_{1,2}(V_{2,3}) = 0.6$	$U_{2,2}(V_{2,3}) = 0.8$
$V_{2,4}$ = BERMUDA SHORTS	$U_{1,2}(V_{2,4}) = 0.4$	$U_{2,2}(V_{2,4}) = 0.5$
Issue 3: shoes	$w_{1,3} = 0.55$	$w_{2,3} = 0.102$
$V_{3,1}$ = SNEAKERS	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = BOOTS	$U_{1,3}(V_{3,2}) = 0.5$	$U_{2,3}(V_{3,2}) = 0.833$
$V_{3,3}$ = SLIPPERS	$U_{1,3}(V_{3,3}) = 0.033$	$U_{2,3}(V_{3,3}) = 0.333$
$V_{3,4}$ = SANDALS	$U_{1,3}(V_{3,4}) = 0.667$	$U_{2,3}(V_{3,4}) = 0.667$
Issue 4: accessories	$w_{1,4} = 0.099$	$w_{2,4} = 0.201$
$V_{4,1}$ = HAT	$U_{1,4}(V_{4,1}) = 0.5$	$U_{2,4}(V_{4,1}) = 0.125$
$V_{4,2}$ = SUNGLASSES	$U_{1,4}(V_{4,2}) = 1$	$U_{2,4}(V_{4,2}) = 1$

TABLE A.27: Outfit scenario specification.

Rank	Agent	Score
1-2	TheNegotiator Reloaded	0.668 ±0.004
1-2	CUHKAgent	0.668 ±0.005
3-4	OMACagent	0.651 ±0.007
3-7	IAMhaggler2012	0.643 ±0.007
4-7	AgentLG	0.641 ±0.004
4-7	Meta-Agent	0.635 ±0.003
4-7	BRAMAgent2	0.634 ±0.004
8	AgentMR	0.327 ±0.003

Table A.28: Scores in the Outfit scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

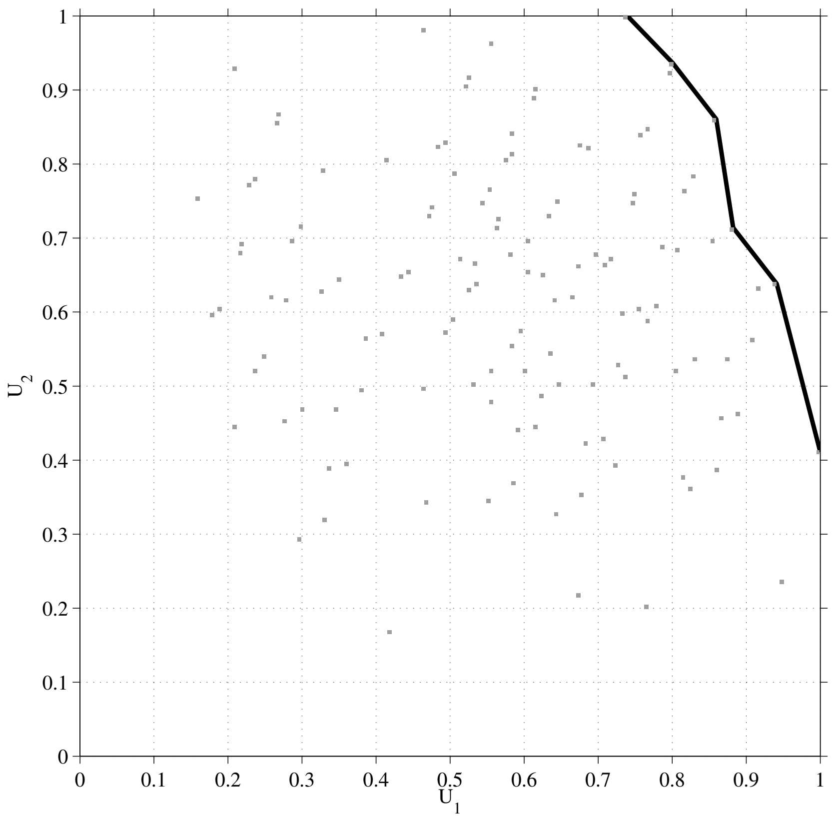


Figure A.14: Outfit scenario outcome space.

A.15 Phone Scenario

Domain	Profile 1	Profile 2
Issue 1: Brand	$w_{1,1} = 0.302$	$w_{2,1} = 0.198$
$V_{1,1}$ = HP	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2}$ = MOTOROLA	$U_{1,1}(V_{1,2}) = 0.2$	$U_{2,1}(V_{1,2}) = 0.5$
$V_{1,3}$ = NOKIA	$U_{1,1}(V_{1,3}) = 0.1$	$U_{2,1}(V_{1,3}) = 0.2$
$V_{1,4}$ = APPLE	$U_{1,1}(V_{1,4}) = 0.5$	$U_{2,1}(V_{1,4}) = 0.1$
Issue 2: Color	$w_{1,2} = 0.204$	$w_{2,2} = 0.046$
$V_{2,1}$ = WHITE	$U_{1,2}(V_{2,1}) = 0.1$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = GREY	$U_{1,2}(V_{2,2}) = 0.5$	$U_{2,2}(V_{2,2}) = 0.3$
$V_{2,3}$ = BLACK	$U_{1,2}(V_{2,3}) = 0.3$	$U_{2,2}(V_{2,3}) = 0.1$
$V_{2,4}$ = SILVER	$U_{1,2}(V_{2,4}) = 0.3$	$U_{2,2}(V_{2,4}) = 0.2$
$V_{2,5}$ = RED	$U_{1,2}(V_{2,5}) = 1$	$U_{2,2}(V_{2,5}) = 0.5$
Issue 3: Operating System	$w_{1,3} = 0.053$	$w_{2,3} = 0.497$
$V_{3,1}$ = WINDOWS MOBILE	$U_{1,3}(V_{3,1}) = 0.1$	$U_{2,3}(V_{3,1}) = 0.2$
$V_{3,2}$ = ANDROID	$U_{1,3}(V_{3,2}) = 0.2$	$U_{2,3}(V_{3,2}) = 1$
$V_{3,3}$ = APPLE	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.1$
$V_{3,4}$ = BLACKBERRY	$U_{1,3}(V_{3,4}) = 1$	$U_{2,3}(V_{3,4}) = 0.5$
Issue 4: Memory	$w_{1,4} = 0.153$	$w_{2,4} = 0.152$
$V_{4,1}$ = 510M	$U_{1,4}(V_{4,1}) = 0.1$	$U_{2,4}(V_{4,1}) = 0.1$
$V_{4,2}$ = 1G	$U_{1,4}(V_{4,2}) = 0.2$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3}$ = 2G	$U_{1,4}(V_{4,3}) = 1$	$U_{2,4}(V_{4,3}) = 0.2$
$V_{4,4}$ = 4G	$U_{1,4}(V_{4,4}) = 0.5$	$U_{2,4}(V_{4,4}) = 0.5$
Issue 5: Screen Resolution	$w_{1,5} = 0.3$	$w_{2,5} = 0.1$
$V_{5,1}$ = 240*400	$U_{1,5}(V_{5,1}) = 0.1$	$U_{2,5}(V_{5,1}) = 0.1$
$V_{5,2}$ = 480*800	$U_{1,5}(V_{5,2}) = 0.2$	$U_{2,5}(V_{5,2}) = 0.3$
$V_{5,3}$ = 600*1024	$U_{1,5}(V_{5,3}) = 0.5$	$U_{2,5}(V_{5,3}) = 0.5$
$V_{5,4}$ = 800*1280	$U_{1,5}(V_{5,4}) = 0.3$	$U_{2,5}(V_{5,4}) = 1$
$V_{5,5}$ = 1280*1280	$U_{1,5}(V_{5,5}) = 1$	$U_{2,5}(V_{5,5}) = 0.2$

TABLE A.29: Phone scenario specification.

Rank	Agent	Score
1	CUHKAgent	0.717 ±0.002
2	TheNegotiator Reloaded	0.704 ±0.003
3-5	AgentLG	0.689 ±0.004
3-5	OMACagent	0.688 ±0.000
3-5	IAMhaggler2012	0.683 ±0.004
6	Meta-Agent	0.674 ±0.002
7	BRAMAgent2	0.663 ±0.004
8	AgentMR	0.341 ±0.002

TABLE A.30: Scores in the Phone scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

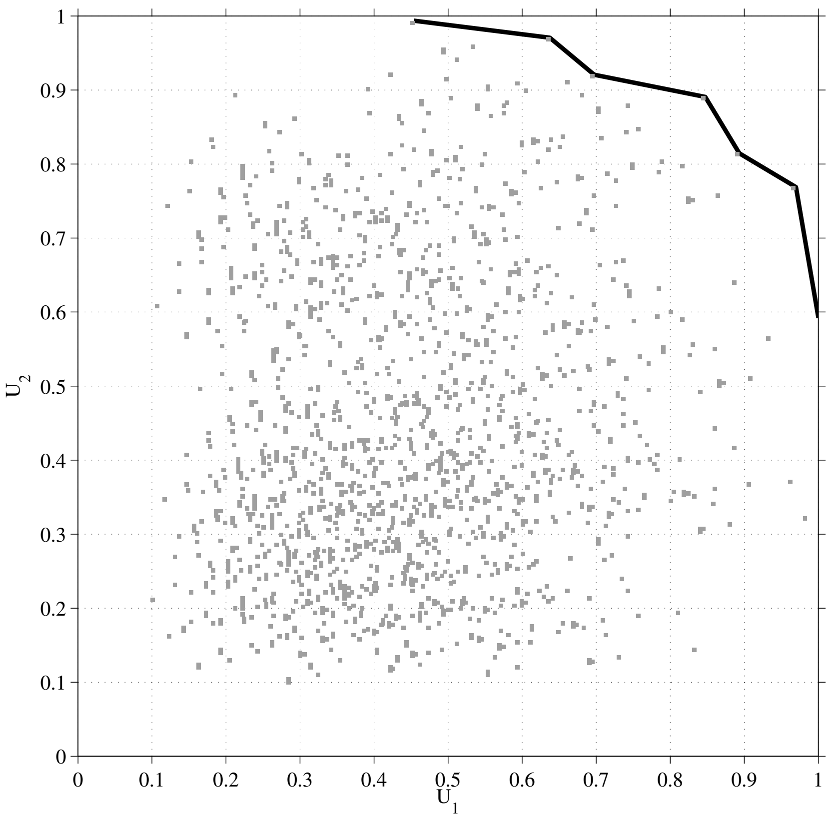


FIGURE A.15: Phone scenario outcome space.

A.16 Rental House Scenario

Domain	Profile 1	Profile 2
Issue 1: price	$w_{1,1} = 0.25$	$w_{2,1} = 0.4$
$V_{1,1} = 40000$YEN	$U_{1,1}(V_{1,1}) = 0.2$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2} = 45000$YEN	$U_{1,1}(V_{1,2}) = 0.4$	$U_{2,1}(V_{1,2}) = 0.8$
$V_{1,3} = 50000$YEN	$U_{1,1}(V_{1,3}) = 0.6$	$U_{2,1}(V_{1,3}) = 0.6$
$V_{1,4} = 55000$YEN	$U_{1,1}(V_{1,4}) = 0.8$	$U_{2,1}(V_{1,4}) = 0.4$
$V_{1,5} = 60000$YEN	$U_{1,1}(V_{1,5}) = 1$	$U_{2,1}(V_{1,5}) = 0.2$
Issue 2: Style	$w_{1,2} = 0.25$	$w_{2,2} = 0.25$
$V_{2,1} =$ JAPANESE	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0.5$
$V_{2,2} =$ WESTERN	$U_{1,2}(V_{2,2}) = 0.5$	$U_{2,2}(V_{2,2}) = 1$
Issue 3: AcceptableLocations	$w_{1,3} = 0.25$	$w_{2,3} = 0.25$
$V_{3,1} =$ NEAR TO FUKIAGE STATION	$U_{1,3}(V_{3,1}) = 0.333$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2} =$ NEAR TO TSRUMAI STATION	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.667$
$V_{3,3} =$ NEAR TO GOKISO STATION	$U_{1,3}(V_{3,3}) = 0.667$	$U_{2,3}(V_{3,3}) = 0.333$
Issue 4: WaterHeaterType	$w_{1,4} = 0.25$	$w_{2,4} = 0.1$
$V_{4,1} =$ ELECTRIC	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0.5$
$V_{4,2} =$ GAS	$U_{1,4}(V_{4,2}) = 0.5$	$U_{2,4}(V_{4,2}) = 1$

TABLE A.31: Rental House scenario specification.

Rank	Agent	Score
1-2	AgentLG	0.581 ±0.006
1-3	OMACagent	0.575 ±0.003
2-3	Meta-Agent	0.572 ±0.005
4-5	TheNegotiator Reloaded	0.564 ±0.001
4-7	CUHKAgent	0.557 ±0.005
5-7	IAMhaggler2012	0.552 ±0.001
5-7	BRAMAgent2	0.551 ±0.004
8	AgentMR	0.343 ±0.001

TABLE A.32: Scores in the Rental House scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

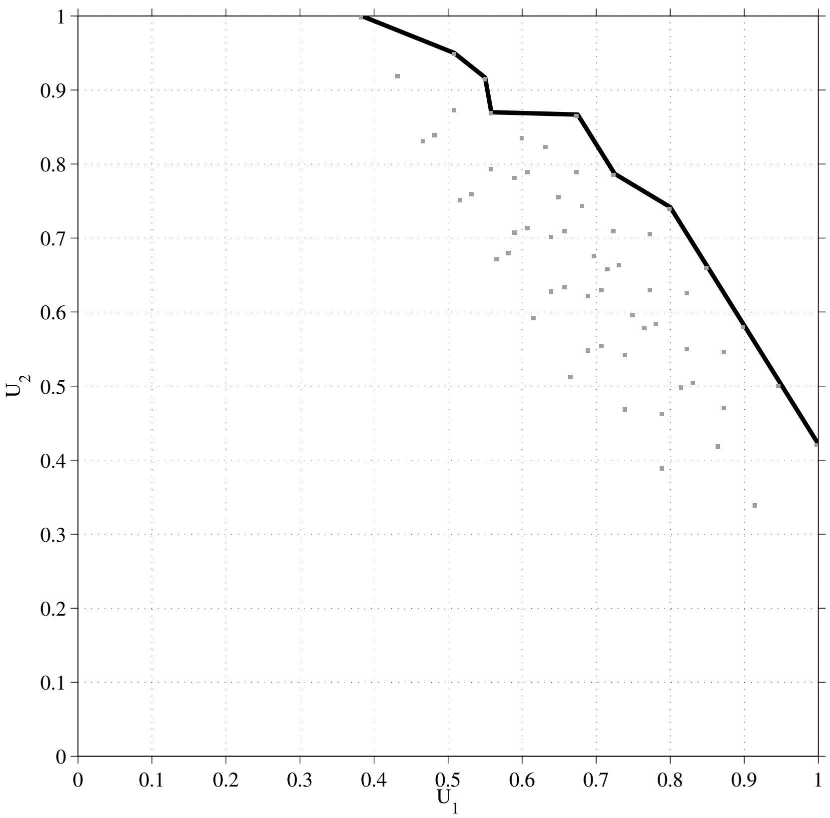


FIGURE A.16: Rental House scenario outcome space.

A.17 Supermarket Scenario

Domain	Profile 1	Profile 2
Issue 1: Bread type	$w_{1,1} = 0.3$	$w_{2,1} = 0.25$
$V_{1,1}$ = Baguette	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.5$
$V_{1,2}$ = Crackers	$U_{1,1}(V_{1,2}) = 0.2$	$U_{2,1}(V_{1,2}) = 0.1$
$V_{1,3}$ = Croissants	$U_{1,1}(V_{1,3}) = 0.1$	$U_{2,1}(V_{1,3}) = 0.2$
$V_{1,4}$ = Plain bread	$U_{1,1}(V_{1,4}) = 0.5$	$U_{2,1}(V_{1,4}) = 1$
Issue 2: Fruit	$w_{1,2} = 0.2$	$w_{2,2} = 0.2$
$V_{2,1}$ = Apples	$U_{1,2}(V_{2,1}) = 0.1$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = Bananas	$U_{1,2}(V_{2,2}) = 0.5$	$U_{2,2}(V_{2,2}) = 0.3$
$V_{2,3}$ = Cherries	$U_{1,2}(V_{2,3}) = 0.3$	$U_{2,2}(V_{2,3}) = 0.3$
$V_{2,4}$ = Grapes	$U_{1,2}(V_{2,4}) = 0.3$	$U_{2,2}(V_{2,4}) = 0.5$
$V_{2,5}$ = Oranges	$U_{1,2}(V_{2,5}) = 0.3$	$U_{2,2}(V_{2,5}) = 0.4$
$V_{2,6}$ = Melons	$U_{1,2}(V_{2,6}) = 0.8$	$U_{2,2}(V_{2,6}) = 0.4$
$V_{2,7}$ = Strawberries	$U_{1,2}(V_{2,7}) = 0.9$	$U_{2,2}(V_{2,7}) = 0.7$
Issue 3: Snacks	$w_{1,3} = 0.05$	$w_{2,3} = 0.05$
$V_{3,1}$ = Chocolate bars	$U_{1,3}(V_{3,1}) = 0.1$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = Doughnuts	$U_{1,3}(V_{3,2}) = 0.2$	$U_{2,3}(V_{3,2}) = 0.5$
$V_{3,3}$ = Nachos	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.2$
$V_{3,4}$ = Popcorn	$U_{1,3}(V_{3,4}) = 1$	$U_{2,3}(V_{3,4}) = 0.1$
$V_{3,5}$ = Potato Chips	$U_{1,3}(V_{3,5}) = 0.1$	$U_{2,3}(V_{3,5}) = 0.7$
$V_{3,6}$ = Candy	$U_{1,3}(V_{3,6}) = 0.7$	$U_{2,3}(V_{3,6}) = 0.3$
$V_{3,7}$ = Cookies	$U_{1,3}(V_{3,7}) = 0.5$	$U_{2,3}(V_{3,7}) = 0.4$
Issue 4: Spreads	$w_{1,4} = 0.15$	$w_{2,4} = 0.15$
$V_{4,1}$ = Cheese	$U_{1,4}(V_{4,1}) = 0.1$	$U_{2,4}(V_{4,1}) = 0.5$
$V_{4,2}$ = Jam	$U_{1,4}(V_{4,2}) = 0.2$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3}$ = Peanut butter	$U_{1,4}(V_{4,3}) = 1$	$U_{2,4}(V_{4,3}) = 0.2$
$V_{4,4}$ = Sandwich spread	$U_{1,4}(V_{4,4}) = 0.5$	$U_{2,4}(V_{4,4}) = 0.1$
$V_{4,5}$ = Chocolate	$U_{1,4}(V_{4,5}) = 0.8$	$U_{2,4}(V_{4,5}) = 0.3$
$V_{4,6}$ = Ham	$U_{1,4}(V_{4,6}) = 0.9$	$U_{2,4}(V_{4,6}) = 0.5$
$V_{4,7}$ = Salami	$U_{1,4}(V_{4,7}) = 0.6$	$U_{2,4}(V_{4,7}) = 0.7$
$V_{4,8}$ = Egg Salad	$U_{1,4}(V_{4,8}) = 0.8$	$U_{2,4}(V_{4,8}) = 0.6$
Issue 5: Vegetables	$w_{1,5} = 0.25$	$w_{2,5} = 0.25$
$V_{5,1}$ = Beans	$U_{1,5}(V_{5,1}) = 0.1$	$U_{2,5}(V_{5,1}) = 1$
$V_{5,2}$ = Broccoli	$U_{1,5}(V_{5,2}) = 0.2$	$U_{2,5}(V_{5,2}) = 0.3$
$V_{5,3}$ = Leek	$U_{1,5}(V_{5,3}) = 0.5$	$U_{2,5}(V_{5,3}) = 0.5$
$V_{5,4}$ = Potatoes	$U_{1,5}(V_{5,4}) = 0.3$	$U_{2,5}(V_{5,4}) = 0.2$
$V_{5,5}$ = Spinach	$U_{1,5}(V_{5,5}) = 1$	$U_{2,5}(V_{5,5}) = 0.1$
$V_{5,6}$ = Carrots	$U_{1,5}(V_{5,6}) = 0.3$	$U_{2,5}(V_{5,6}) = 0.5$
$V_{5,7}$ = Tomatoes	$U_{1,5}(V_{5,7}) = 0.8$	$U_{2,5}(V_{5,7}) = 0.7$
Issue 6: Drinks	$w_{1,6} = 0.05$	$w_{2,6} = 0.1$
$V_{6,1}$ = Energy Drinks	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 1$
$V_{6,2}$ = Milk	$U_{1,6}(V_{6,2}) = 0.7$	$U_{2,6}(V_{6,2}) = 0.3$
$V_{6,3}$ = Tea	$U_{1,6}(V_{6,3}) = 0.2$	$U_{2,6}(V_{6,3}) = 0.5$
$V_{6,4}$ = Coffee	$U_{1,6}(V_{6,4}) = 0.4$	$U_{2,6}(V_{6,4}) = 0.2$
$V_{6,5}$ = Juice	$U_{1,6}(V_{6,5}) = 0.7$	$U_{2,6}(V_{6,5}) = 0.1$
$V_{6,6}$ = Coca Cola	$U_{1,6}(V_{6,6}) = 0.8$	$U_{2,6}(V_{6,6}) = 0.6$
$V_{6,7}$ = Fanta	$U_{1,6}(V_{6,7}) = 0.7$	$U_{2,6}(V_{6,7}) = 0.8$
$V_{6,8}$ = Beer	$U_{1,6}(V_{6,8}) = 0.7$	$U_{2,6}(V_{6,8}) = 0.7$
$V_{6,9}$ = Wine	$U_{1,6}(V_{6,9}) = 0.5$	$U_{2,6}(V_{6,9}) = 0.5$

Table A.33: Supermarket scenario specification.

Rank	Agent	Score
1	AgentLG	0.565 ±0.007
2	OMACagent	0.550 ±0.003
3-4	IAMhaggler2012	0.528 ±0.004
3-4	TheNegotiator Reloaded	0.527 ±0.002
5	CUHKAgent	0.519 ±0.004
6-7	BRAMAgent2	0.472 ±0.010
6-7	Meta-Agent	0.460 ±0.013
8	AgentMR	0.223 ±0.004

TABLE A.34: Scores in the Supermarket scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

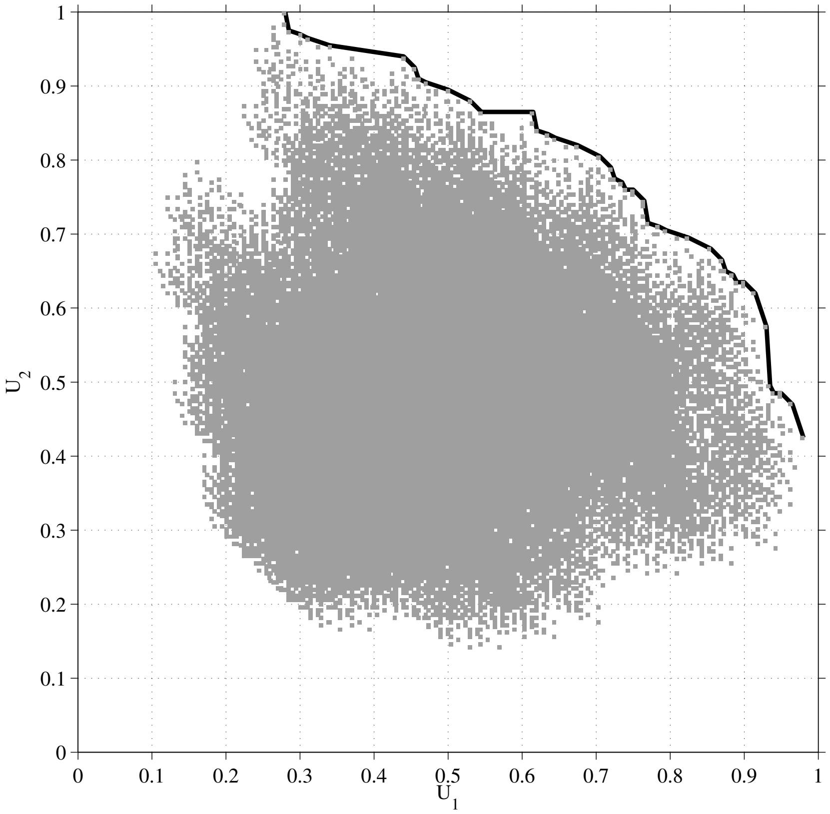


FIGURE A.17: Supermarket scenario outcome space.

A.18 ADG Scenario

Domain	Profile 1	Profile 2
Issue 1: CD player	$w_{1,1} = 0.16$	$w_{2,1} = 0.36$
$V_{1,1} =$ GOOD	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0$
$V_{1,2} =$ FAIRLY GOOD	$U_{1,1}(V_{1,2}) = 0.98$	$U_{2,1}(V_{1,2}) = 0.85$
$V_{1,3} =$ STANDARD	$U_{1,1}(V_{1,3}) = 0.92$	$U_{2,1}(V_{1,3}) = 0.91$
$V_{1,4} =$ MEAGRE	$U_{1,1}(V_{1,4}) = 0.75$	$U_{2,1}(V_{1,4}) = 0.98$
$V_{1,5} =$ NONE	$U_{1,1}(V_{1,5}) = 0$	$U_{2,1}(V_{1,5}) = 1$
Issue 2: Extra speakers	$w_{1,2} = 0.16$	$w_{2,2} = 0.2$
$V_{2,1} =$ GOOD	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0$
$V_{2,2} =$ FAIRLY GOOD	$U_{1,2}(V_{2,2}) = 0.99$	$U_{2,2}(V_{2,2}) = 0.8$
$V_{2,3} =$ STANDARD	$U_{1,2}(V_{2,3}) = 0.93$	$U_{2,2}(V_{2,3}) = 0.96$
$V_{2,4} =$ MEAGRE	$U_{1,2}(V_{2,4}) = 0.8$	$U_{2,2}(V_{2,4}) = 0.93$
$V_{2,5} =$ NONE	$U_{1,2}(V_{2,5}) = 0$	$U_{2,2}(V_{2,5}) = 1$
Issue 3: Air conditioning	$w_{1,3} = 0.17$	$w_{2,3} = 0.2$
$V_{3,1} =$ GOOD	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0$
$V_{3,2} =$ FAIRLY GOOD	$U_{1,3}(V_{3,2}) = 0.99$	$U_{2,3}(V_{3,2}) = 0.8$
$V_{3,3} =$ STANDARD	$U_{1,3}(V_{3,3}) = 0.96$	$U_{2,3}(V_{3,3}) = 0.91$
$V_{3,4} =$ MEAGRE	$U_{1,3}(V_{3,4}) = 0.83$	$U_{2,3}(V_{3,4}) = 0.99$
$V_{3,5} =$ NONE	$U_{1,3}(V_{3,5}) = 0$	$U_{2,3}(V_{3,5}) = 1$
Issue 4: Tow hedge	$w_{1,4} = 0.17$	$w_{2,4} = 0.2$
$V_{4,1} =$ GOOD	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0$
$V_{4,2} =$ FAIRLY GOOD	$U_{1,4}(V_{4,2}) = 0.99$	$U_{2,4}(V_{4,2}) = 0.8$
$V_{4,3} =$ STANDARD	$U_{1,4}(V_{4,3}) = 0.93$	$U_{2,4}(V_{4,3}) = 0.9$
$V_{4,4} =$ MEAGRE	$U_{1,4}(V_{4,4}) = 0.85$	$U_{2,4}(V_{4,4}) = 0.98$
$V_{4,5} =$ NONE	$U_{1,4}(V_{4,5}) = 0$	$U_{2,4}(V_{4,5}) = 1$
Issue 5: Tow hedge2	$w_{1,5} = 0.17$	$w_{2,5} = 0.02$
$V_{5,1} =$ GOOD	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0$
$V_{5,2} =$ FAIRLY GOOD	$U_{1,5}(V_{5,2}) = 0.99$	$U_{2,5}(V_{5,2}) = 0.8$
$V_{5,3} =$ STANDARD	$U_{1,5}(V_{5,3}) = 0.95$	$U_{2,5}(V_{5,3}) = 0.9$
$V_{5,4} =$ MEAGRE	$U_{1,5}(V_{5,4}) = 0.8$	$U_{2,5}(V_{5,4}) = 0.95$
$V_{5,5} =$ NONE	$U_{1,5}(V_{5,5}) = 0$	$U_{2,5}(V_{5,5}) = 1$
Issue 6: Tow hedge3	$w_{1,6} = 0.17$	$w_{2,6} = 0.02$
$V_{6,1} =$ GOOD	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 0$
$V_{6,2} =$ FAIRLY GOOD	$U_{1,6}(V_{6,2}) = 0.99$	$U_{2,6}(V_{6,2}) = 0.8$
$V_{6,3} =$ STANDARD	$U_{1,6}(V_{6,3}) = 0.95$	$U_{2,6}(V_{6,3}) = 0.9$
$V_{6,4} =$ MEAGRE	$U_{1,6}(V_{6,4}) = 0.8$	$U_{2,6}(V_{6,4}) = 0.95$
$V_{6,5} =$ NONE	$U_{1,6}(V_{6,5}) = 0$	$U_{2,6}(V_{6,5}) = 1$

TABLE A.35: ADG scenario specification.

Rank	Agent	Score
1	AgentLG	0.775 ±0.002
2-3	CUHKAgent	0.764 ±0.006
2-3	TheNegotiator Reloaded	0.759 ±0.003
4	BRAMAgent2	0.751 ±0.002
5	IAMhaggler2012	0.733 ±0.003
6	Meta-Agent	0.721 ±0.002
7	OMACagent	0.710 ±0.005
8	AgentMR	0.188 ±0.000

TABLE A.36: Scores in the ADG scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

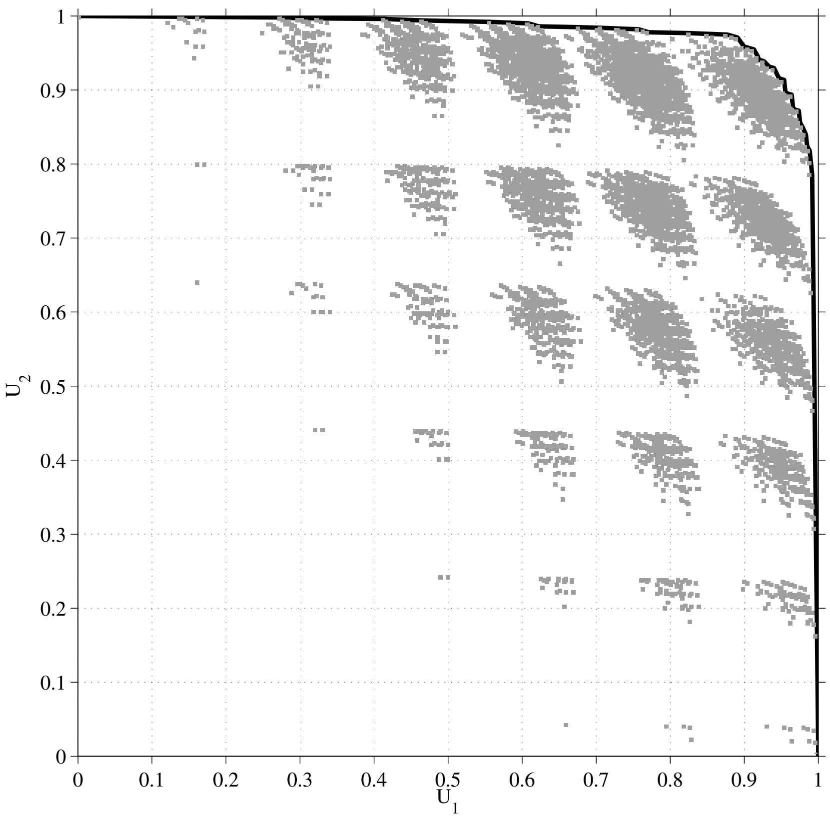


FIGURE A.18: ADG scenario outcome space.

A.19 Amsterdam Party Scenario

Domain	Profile 1	Profile 2
Issue 1: Venue	$w_{1,1} = 0.18$	$w_{2,1} = 0.15$
$V_{1,1} =$ Shopping	$U_{1,1}(V_{1,1}) = 0.5$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2} =$ Museum	$U_{1,1}(V_{1,2}) = 1$	$U_{2,1}(V_{1,2}) = 0.25$
$V_{1,3} =$ Dancing	$U_{1,1}(V_{1,3}) = 0.5$	$U_{2,1}(V_{1,3}) = 0.5$
$V_{1,4} =$ Diner	$U_{1,1}(V_{1,4}) = 0.25$	$U_{2,1}(V_{1,4}) = 0.75$
Issue 2: Time of arrival	$w_{1,2} = 0.22$	$w_{2,2} = 0.12$
$V_{2,1} =$ Morning	$U_{1,2}(V_{2,1}) = 0.333$	$U_{2,2}(V_{2,1}) = 0.333$
$V_{2,2} =$ Afternoon	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.667$
$V_{2,3} =$ Evening	$U_{1,2}(V_{2,3}) = 0.333$	$U_{2,2}(V_{2,3}) = 1$
Issue 3: Day of the week	$w_{1,3} = 0.13$	$w_{2,3} = 0.21$
$V_{3,1} =$ Monday	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0.167$
$V_{3,2} =$ Tuesday	$U_{1,3}(V_{3,2}) = 0.5$	$U_{2,3}(V_{3,2}) = 0.333$
$V_{3,3} =$ Wednesday	$U_{1,3}(V_{3,3}) = 0.667$	$U_{2,3}(V_{3,3}) = 0.5$
$V_{3,4} =$ Thursday	$U_{1,3}(V_{3,4}) = 0.833$	$U_{2,3}(V_{3,4}) = 0.667$
$V_{3,5} =$ Friday	$U_{1,3}(V_{3,5}) = 0.167$	$U_{2,3}(V_{3,5}) = 0.667$
$V_{3,6} =$ Saturday	$U_{1,3}(V_{3,6}) = 0.167$	$U_{2,3}(V_{3,6}) = 1$
$V_{3,7} =$ Sunday	$U_{1,3}(V_{3,7}) = 1$	$U_{2,3}(V_{3,7}) = 1$
Issue 4: Duration	$w_{1,4} = 0.12$	$w_{2,4} = 0.32$
$V_{4,1} =$ One day	$U_{1,4}(V_{4,1}) = 1$	$U_{2,4}(V_{4,1}) = 0.333$
$V_{4,2} =$ One night	$U_{1,4}(V_{4,2}) = 0.667$	$U_{2,4}(V_{4,2}) = 0.667$
$V_{4,3} =$ One week	$U_{1,4}(V_{4,3}) = 0.333$	$U_{2,4}(V_{4,3}) = 1$
Issue 5: Transportation	$w_{1,5} = 0.23$	$w_{2,5} = 0.13$
$V_{5,1} =$ Public transport	$U_{1,5}(V_{5,1}) = 1$	$U_{2,5}(V_{5,1}) = 0.333$
$V_{5,2} =$ Car	$U_{1,5}(V_{5,2}) = 0.333$	$U_{2,5}(V_{5,2}) = 1$
$V_{5,3} =$ Combination	$U_{1,5}(V_{5,3}) = 0.667$	$U_{2,5}(V_{5,3}) = 0.667$
Issue 6: Souvenirs	$w_{1,6} = 0.12$	$w_{2,6} = 0.07$
$V_{6,1} =$ None	$U_{1,6}(V_{6,1}) = 0.75$	$U_{2,6}(V_{6,1}) = 0.25$
$V_{6,2} =$ Tulips	$U_{1,6}(V_{6,2}) = 1$	$U_{2,6}(V_{6,2}) = 0.75$
$V_{6,3} =$ Cheese	$U_{1,6}(V_{6,3}) = 0.25$	$U_{2,6}(V_{6,3}) = 0$
$V_{6,4} =$ Waterpipe	$U_{1,6}(V_{6,4}) = 0.25$	$U_{2,6}(V_{6,4}) = 1$

Table A.37: Amsterdam Party scenario specification.

Rank	Agent	Score
1	CUHKAgent	0.654 ±0.002
2-3	AgentLG	0.643 ±0.002
2-4	TheNegotiator Reloaded	0.631 ±0.009
3-4	OMACagent	0.628 ±0.003
5-6	BRAMAgent2	0.600 ±0.002
5-6	Meta-Agent	0.599 ±0.004
7	IAMhaggler2012	0.577 ±0.004
8	AgentMR	0.188 ±0.000

Table A.38: Scores in the Amsterdam Party scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

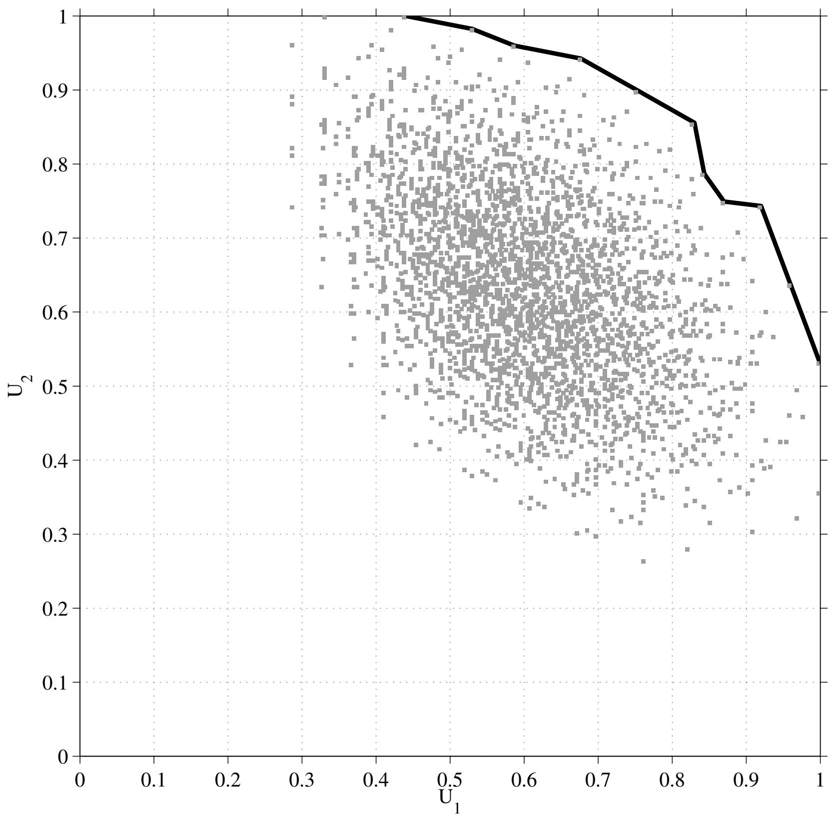


Figure A.19: Amsterdam Party scenario outcome space.

A.20 Grocery Scenario

Domain	Profile 1	Profile 2
Issue 1: Bread type	$w_{1,1} = 0.3$	$w_{2,1} = 0.2$
$V_{1,1} =$ Baguette	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 1$
$V_{1,2} =$ Crackers	$U_{1,1}(V_{1,2}) = 0.2$	$U_{2,1}(V_{1,2}) = 0.5$
$V_{1,3} =$ Croissants	$U_{1,1}(V_{1,3}) = 0.1$	$U_{2,1}(V_{1,3}) = 0.2$
$V_{1,4} =$ Plain bread	$U_{1,1}(V_{1,4}) = 0.5$	$U_{2,1}(V_{1,4}) = 0.1$
Issue 2: Fruit	$w_{1,2} = 0.2$	$w_{2,2} = 0.05$
$V_{2,1} =$ Apples	$U_{1,2}(V_{2,1}) = 0.1$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2} =$ Bananas	$U_{1,2}(V_{2,2}) = 0.5$	$U_{2,2}(V_{2,2}) = 0.3$
$V_{2,3} =$ Cherries	$U_{1,2}(V_{2,3}) = 0.3$	$U_{2,2}(V_{2,3}) = 0.1$
$V_{2,4} =$ Grapes	$U_{1,2}(V_{2,4}) = 0.3$	$U_{2,2}(V_{2,4}) = 0.2$
$V_{2,5} =$ Pears	$U_{1,2}(V_{2,5}) = 1$	$U_{2,2}(V_{2,5}) = 0.5$
Issue 3: Snacks	$w_{1,3} = 0.05$	$w_{2,3} = 0.5$
$V_{3,1} =$ Chocolate bars	$U_{1,3}(V_{3,1}) = 0.1$	$U_{2,3}(V_{3,1}) = 0.2$
$V_{3,2} =$ Doughnuts	$U_{1,3}(V_{3,2}) = 0.2$	$U_{2,3}(V_{3,2}) = 1$
$V_{3,3} =$ Nachos	$U_{1,3}(V_{3,3}) = 0.5$	$U_{2,3}(V_{3,3}) = 0.1$
$V_{3,4} =$ Popcorn	$U_{1,3}(V_{3,4}) = 1$	$U_{2,3}(V_{3,4}) = 0.5$
Issue 4: Spreads	$w_{1,4} = 0.15$	$w_{2,4} = 0.15$
$V_{4,1} =$ Cheese	$U_{1,4}(V_{4,1}) = 0.1$	$U_{2,4}(V_{4,1}) = 0.1$
$V_{4,2} =$ Jam	$U_{1,4}(V_{4,2}) = 0.2$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3} =$ Peanut butter	$U_{1,4}(V_{4,3}) = 1$	$U_{2,4}(V_{4,3}) = 0.2$
$V_{4,4} =$ Sandwich spread	$U_{1,4}(V_{4,4}) = 0.5$	$U_{2,4}(V_{4,4}) = 0.5$
Issue 5: Vegetables	$w_{1,5} = 0.3$	$w_{2,5} = 0.1$
$V_{5,1} =$ Beans	$U_{1,5}(V_{5,1}) = 0.1$	$U_{2,5}(V_{5,1}) = 0.1$
$V_{5,2} =$ Broccoli	$U_{1,5}(V_{5,2}) = 0.2$	$U_{2,5}(V_{5,2}) = 0.3$
$V_{5,3} =$ Leek	$U_{1,5}(V_{5,3}) = 0.5$	$U_{2,5}(V_{5,3}) = 0.5$
$V_{5,4} =$ Potatoes	$U_{1,5}(V_{5,4}) = 0.3$	$U_{2,5}(V_{5,4}) = 1$
$V_{5,5} =$ Spinach	$U_{1,5}(V_{5,5}) = 1$	$U_{2,5}(V_{5,5}) = 0.2$

Table A.39: Grocery scenario specification.

Rank	Agent	Score
1-2	CUHKAgent	0.723 ±0.002
1-3	TheNegotiator Reloaded	0.720 ±0.006
2-3	AgentLG	0.712 ±0.007
4-6	OMACagent	0.692 ±0.006
4-7	BRAMAgent2	0.690 ±0.004
4-7	Meta-Agent	0.685 ±0.006
5-7	IAMhaggler2012	0.679 ±0.008
8	AgentMR	0.458 ±0.002

TABLE A.40: Scores in the Grocery scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

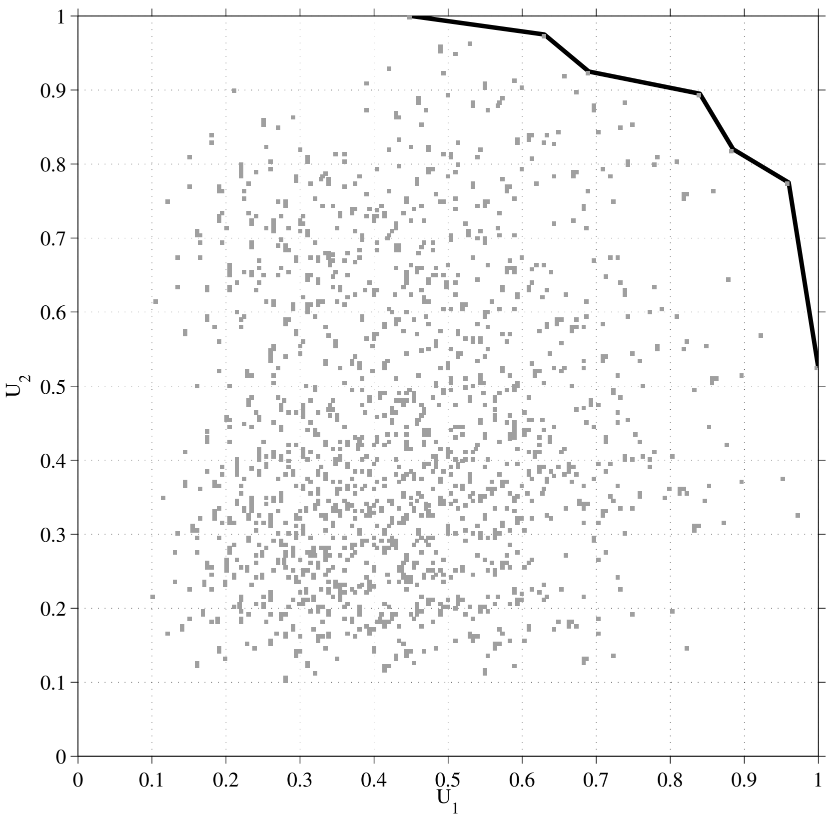


FIGURE A.20: Grocery scenario outcome space.

A.21 Laptop Scenario

Domain	Profile 1	Profile 2
Issue 1: Laptop	$w_{1,1} = 0.445$	$w_{2,1} = 0.378$
$V_{1,1}$ = Dell	$U_{1,1}(V_{1,1}) = 0.4$	$U_{2,1}(V_{1,1}) = 0.4$
$V_{1,2}$ = Macintosh	$U_{1,1}(V_{1,2}) = 0.667$	$U_{2,1}(V_{1,2}) = 1$
$V_{1,3}$ = HP	$U_{1,1}(V_{1,3}) = 1$	$U_{2,1}(V_{1,3}) = 0.667$
Issue 2: Harddisk	$w_{1,2} = 0.378$	$w_{2,2} = 0.177$
$V_{2,1}$ = 60 Gb	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 0.667$
$V_{2,2}$ = 80 Gb	$U_{1,2}(V_{2,2}) = 0.667$	$U_{2,2}(V_{2,2}) = 1$
$V_{2,3}$ = 120 Gb	$U_{1,2}(V_{2,3}) = 0.3$	$U_{2,2}(V_{2,3}) = 0.3$
Issue 3: External Monitor	$w_{1,3} = 0.177$	$w_{2,3} = 0.445$
$V_{3,1}$ = 19 LCD	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2}$ = 20 LCD	$U_{1,3}(V_{3,2}) = 0.333$	$U_{2,3}(V_{3,2}) = 0.667$
$V_{3,3}$ = 23 LCD	$U_{1,3}(V_{3,3}) = 0.667$	$U_{2,3}(V_{3,3}) = 0.333$

Table A.41: Laptop scenario specification.

Rank	Agent	Score
1-2	CUHKAgent	0.775 ±0.007
1-2	TheNegotiator Reloaded	0.773 ±0.001
3-4	IAMhaggler2012	0.760 ±0.008
3-5	Meta-Agent	0.751 ±0.002
4-5	OMACagent	0.749 ±0.004
6	AgentLG	0.732 ±0.001
7	BRAMAgent2	0.720 ±0.002
8	AgentMR	0.532 ±0.005

TABLE A.42: Scores in the Laptop scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

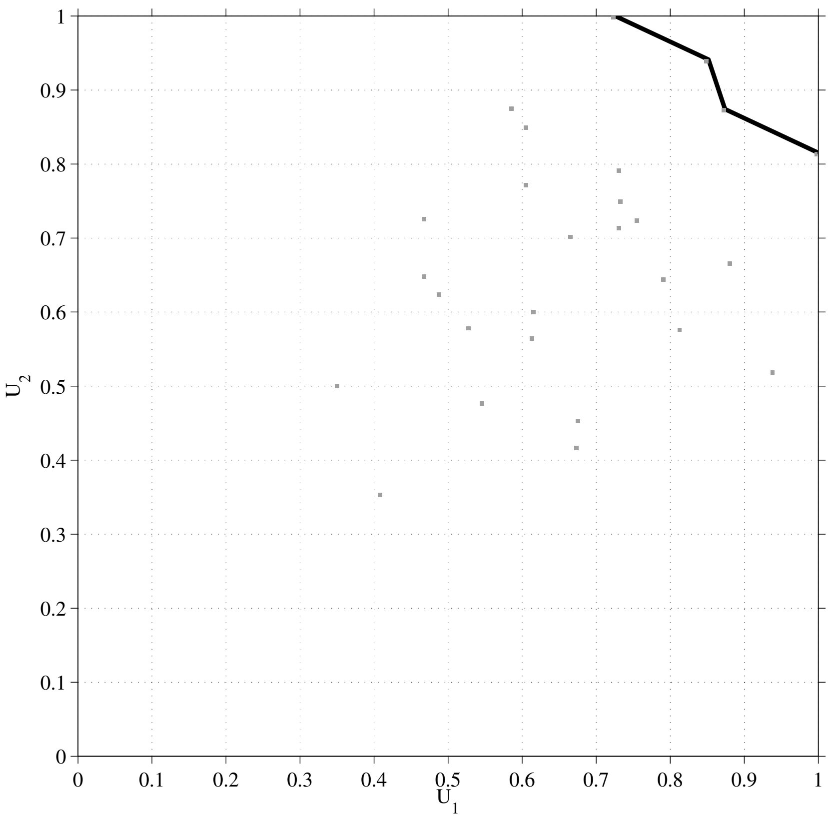


FIGURE A.21: Laptop scenario outcome space.

A.22 NiceOrDie Scenario

Domain	Profile 1	Profile 2
Issue 1: NiceOrDie	$w_{1,1} = 1$	$w_{2,1} = 1$
$V_{1,1} = 100\_16$	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.16$
$V_{1,2} = 16\_100$	$U_{1,1}(V_{1,2}) = 0.16$	$U_{2,1}(V_{1,2}) = 1$
$V_{1,3} = 29\_29$	$U_{1,1}(V_{1,3}) = 0.299$	$U_{2,1}(V_{1,3}) = 0.299$

TABLE A.43: NiceOrDie scenario specification.

Rank	Agent	Score
1-3	BRAMAgent2	0.398 ±0.000
1-3	OMACagent	0.396 ±0.006
1-4	AgentLG	0.391 ±0.006
3-4	IAMhaggler2012	0.386 ±0.000
5	CUHKAgent	0.364 ±0.006
6	Meta-Agent	0.315 ±0.000
7	TheNegotiator Reloaded	0.264 ±0.000
8	AgentMR	0.157 ±0.000

TABLE A.44: Scores in the NiceOrDie scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

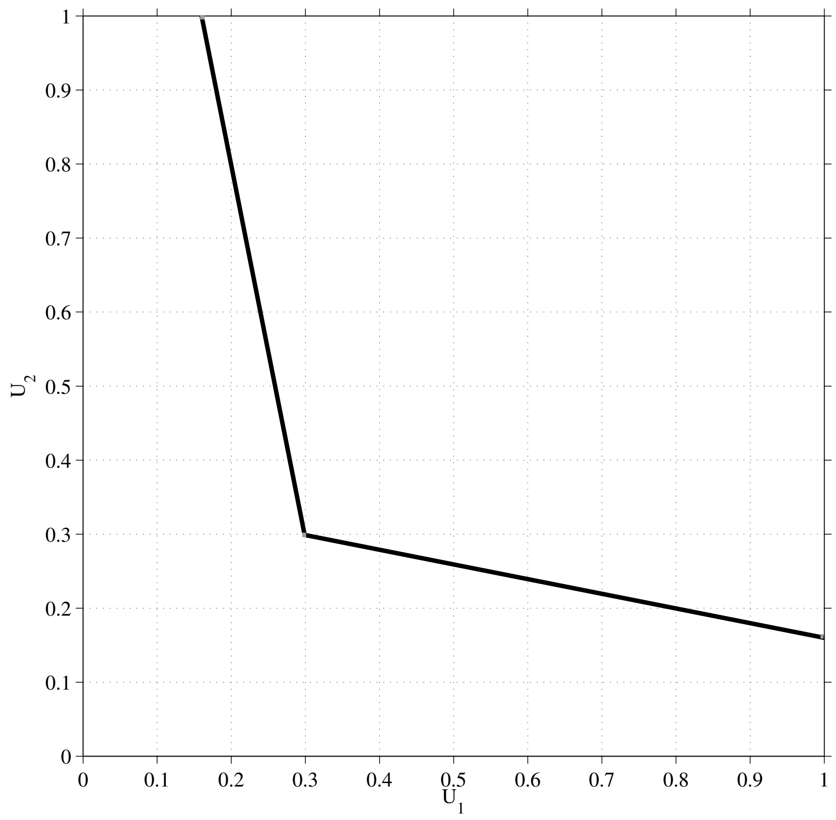


FIGURE A.22: NiceOrDie scenario outcome space.

A.23 Itex vs Cypress Scenario

Domain	Profile 1	Profile 2
Issue 1: Price	$w_{1,1} = 0.288$	$w_{2,1} = 0.47$
$V_{1,1} = \$4.37$	$U_{1,1}(V_{1,1}) = 1$	$U_{2,1}(V_{1,1}) = 0.025$
$V_{1,2} = \$4.12$	$U_{1,1}(V_{1,2}) = 0.667$	$U_{2,1}(V_{1,2}) = 0.25$
$V_{1,3} = \$3.98$	$U_{1,1}(V_{1,3}) = 0.333$	$U_{2,1}(V_{1,3}) = 0.625$
$V_{1,4} = \$3.71$	$U_{1,1}(V_{1,4}) = 0.167$	$U_{2,1}(V_{1,4}) = 0.825$
$V_{1,5} = \$3.47$	$U_{1,1}(V_{1,5}) = 0.033$	$U_{2,1}(V_{1,5}) = 1$
Issue 2: Delivery	$w_{1,2} = 0.192$	$w_{2,2} = 0.122$
$V_{2,1} =$ 60 DAYS	$U_{1,2}(V_{2,1}) = 0.05$	$U_{2,2}(V_{2,1}) = 0.04$
$V_{2,2} =$ 45 DAYS	$U_{1,2}(V_{2,2}) = 1$	$U_{2,2}(V_{2,2}) = 0.4$
$V_{2,3} =$ 30 DAYS	$U_{1,2}(V_{2,3}) = 0.5$	$U_{2,2}(V_{2,3}) = 0.76$
$V_{2,4} =$ 20 DAYS	$U_{1,2}(V_{2,4}) = 0.25$	$U_{2,2}(V_{2,4}) = 1$
Issue 3: Payment	$w_{1,3} = 0.242$	$w_{2,3} = 0.177$
$V_{3,1} =$ UPON DELIVERY	$U_{1,3}(V_{3,1}) = 0.4$	$U_{2,3}(V_{3,1}) = 1$
$V_{3,2} =$ 30 DAYS AFTER DELIVERY	$U_{1,3}(V_{3,2}) = 1$	$U_{2,3}(V_{3,2}) = 0.4$
$V_{3,3} =$ 60 DAYS AFTER DELIVERY	$U_{1,3}(V_{3,3}) = 0.04$	$U_{2,3}(V_{3,3}) = 0.067$
Issue 4: Returns	$w_{1,4} = 0.278$	$w_{2,4} = 0.231$
$V_{4,1} =$ FULL PRICE	$U_{1,4}(V_{4,1}) = 0.033$	$U_{2,4}(V_{4,1}) = 1$
$V_{4,2} =$ 5% SPOILAGE ALLOWED	$U_{1,4}(V_{4,2}) = 1$	$U_{2,4}(V_{4,2}) = 0.35$
$V_{4,3} =$ 10% SPOILAGE ALLOWED	$U_{1,4}(V_{4,3}) = 0.167$	$U_{2,4}(V_{4,3}) = 0.05$

TABLE A.45: Itex vs Cypress scenario specification.

Rank	Agent	Score
1	OMACagent	0.521 ±0.002
2	AgentLG	0.515 ±0.004
3	TheNegotiator Reloaded	0.486 ±0.002
4-5	IAMhaggler2012	0.481 ±0.003
4-7	CUHKAgent	0.475 ±0.006
5-7	BRAMAgent2	0.473 ±0.004
5-7	Meta-Agent	0.471 ±0.002
8	AgentMR	0.306 ±0.003

TABLE A.46: Scores in the Itex vs Cypress scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

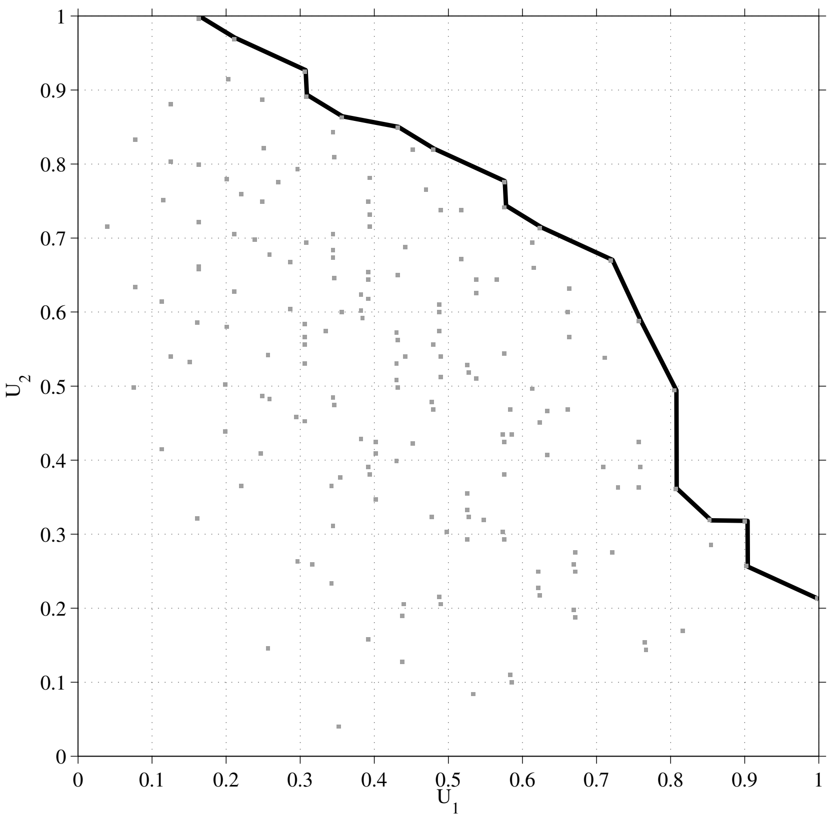


FIGURE A.23: Itex vs Cypress scenario outcome space.

A.24 Travel Scenario

Domain	Profile 1	Profile 2
Issue 1: Atmosphere	$w_{1,1} = 0.37$	$w_{2,1} = 0.212$
$V_{1,1}$ = Cultural heritage	$U_{1,1}(V_{1,1}) = 0.111$	$U_{2,1}(V_{1,1}) = 0.109$
$V_{1,2}$ = Local traditions	$U_{1,1}(V_{1,2}) = 0.089$	$U_{2,1}(V_{1,2}) = 0.109$
$V_{1,3}$ = Political stability	$U_{1,1}(V_{1,3}) = 0.576$	$U_{2,1}(V_{1,3}) = 0.088$
$V_{1,4}$ = Security (personal)	$U_{1,1}(V_{1,4}) = 0.708$	$U_{2,1}(V_{1,4}) = 0.088$
$V_{1,5}$ = Liveliness	$U_{1,1}(V_{1,5}) = 0.122$	$U_{2,1}(V_{1,5}) = 0.121$
$V_{1,6}$ = Touristic activities	$U_{1,1}(V_{1,6}) = 0.1$	$U_{2,1}(V_{1,6}) = 1$
$V_{1,7}$ = Hospitality	$U_{1,1}(V_{1,7}) = 1$	$U_{2,1}(V_{1,7}) = 0.577$
Issue 2: Amusement	$w_{1,2} = 0.043$	$w_{2,2} = 0.037$
$V_{2,1}$ = Nightlife and entertainment	$U_{1,2}(V_{2,1}) = 1$	$U_{2,2}(V_{2,1}) = 1$
$V_{2,2}$ = Nightclubs	$U_{1,2}(V_{2,2}) = 0.089$	$U_{2,2}(V_{2,2}) = 0.576$
$V_{2,3}$ = Excursion	$U_{1,2}(V_{2,3}) = 0.1$	$U_{2,2}(V_{2,3}) = 0.087$
$V_{2,4}$ = Casinos	$U_{1,2}(V_{2,4}) = 0.122$	$U_{2,2}(V_{2,4}) = 0.098$
$V_{2,5}$ = Zoo	$U_{1,2}(V_{2,5}) = 0.708$	$U_{2,2}(V_{2,5}) = 0.111$
$V_{2,6}$ = Festivals	$U_{1,2}(V_{2,6}) = 0.111$	$U_{2,2}(V_{2,6}) = 0.576$
$V_{2,7}$ = Amusement park	$U_{1,2}(V_{2,7}) = 0.576$	$U_{2,2}(V_{2,7}) = 0.121$
Issue 3: Culinary	$w_{1,3} = 0.208$	$w_{2,3} = 0.261$
$V_{3,1}$ = Local cuisine	$U_{1,3}(V_{3,1}) = 1$	$U_{2,3}(V_{3,1}) = 0.708$
$V_{3,2}$ = Lunch facilities	$U_{1,3}(V_{3,2}) = 0.138$	$U_{2,3}(V_{3,2}) = 1$
$V_{3,3}$ = International cuisine	$U_{1,3}(V_{3,3}) = 0.709$	$U_{2,3}(V_{3,3}) = 1$
$V_{3,4}$ = Cooking workshops	$U_{1,3}(V_{3,4}) = 0.122$	$U_{2,3}(V_{3,4}) = 0.119$
Issue 4: Shopping	$w_{1,4} = 0.043$	$w_{2,4} = 0.043$
$V_{4,1}$ = Shopping malls	$U_{1,4}(V_{4,1}) = 0.12$	$U_{2,4}(V_{4,1}) = 0.709$
$V_{4,2}$ = Markets	$U_{1,4}(V_{4,2}) = 0.138$	$U_{2,4}(V_{4,2}) = 1$
$V_{4,3}$ = Streets	$U_{1,4}(V_{4,3}) = 0.709$	$U_{2,4}(V_{4,3}) = 0.12$
$V_{4,4}$ = Small boutiques	$U_{1,4}(V_{4,4}) = 1$	$U_{2,4}(V_{4,4}) = 0.138$
Issue 5: Culture	$w_{1,5} = 0.047$	$w_{2,5} = 0.047$
$V_{5,1}$ = Museum	$U_{1,5}(V_{5,1}) = 0.709$	$U_{2,5}(V_{5,1}) = 0.12$
$V_{5,2}$ = Music hall	$U_{1,5}(V_{5,2}) = 0.12$	$U_{2,5}(V_{5,2}) = 0.709$
$V_{5,3}$ = Theater	$U_{1,5}(V_{5,3}) = 1$	$U_{2,5}(V_{5,3}) = 0.11$
$V_{5,4}$ = Art gallery	$U_{1,5}(V_{5,4}) = 0.099$	$U_{2,5}(V_{5,4}) = 0.579$
$V_{5,5}$ = Cinema	$U_{1,5}(V_{5,5}) = 0.579$	$U_{2,5}(V_{5,5}) = 1$
$V_{5,6}$ = Congress center	$U_{1,5}(V_{5,6}) = 0.11$	$U_{2,5}(V_{5,6}) = 0.099$
Issue 6: Sport	$w_{1,6} = 0.033$	$w_{2,6} = 0.028$
$V_{6,1}$ = Bike tours	$U_{1,6}(V_{6,1}) = 1$	$U_{2,6}(V_{6,1}) = 1$
$V_{6,2}$ = Hiking	$U_{1,6}(V_{6,2}) = 0.709$	$U_{2,6}(V_{6,2}) = 0.11$
$V_{6,3}$ = Indoor activities	$U_{1,6}(V_{6,3}) = 0.11$	$U_{2,6}(V_{6,3}) = 0.137$
$V_{6,4}$ = Outdoor activities	$U_{1,6}(V_{6,4}) = 0.121$	$U_{2,6}(V_{6,4}) = 0.709$
$V_{6,5}$ = Adventure	$U_{1,6}(V_{6,5}) = 0.137$	$U_{2,6}(V_{6,5}) = 0.121$
Issue 7: Environment	$w_{1,7} = 0.255$	$w_{2,7} = 0.371$
$V_{7,1}$ = Parks and Gardens	$U_{1,7}(V_{7,1}) = 0.5$	$U_{2,7}(V_{7,1}) = 0.098$
$V_{7,2}$ = Square	$U_{1,7}(V_{7,2}) = 0.089$	$U_{2,7}(V_{7,2}) = 1$
$V_{7,3}$ = Historical places	$U_{1,7}(V_{7,3}) = 0.098$	$U_{2,7}(V_{7,3}) = 0.708$
$V_{7,4}$ = See, river, etc.	$U_{1,7}(V_{7,4}) = 0.082$	$U_{2,7}(V_{7,4}) = 0.576$
$V_{7,5}$ = Monuments	$U_{1,7}(V_{7,5}) = 1$	$U_{2,7}(V_{7,5}) = 0.708$
$V_{7,6}$ = Special streets	$U_{1,7}(V_{7,6}) = 0.576$	$U_{2,7}(V_{7,6}) = 0.083$
$V_{7,7}$ = Palace	$U_{1,7}(V_{7,7}) = 0.709$	$U_{2,7}(V_{7,7}) = 0.11$
$V_{7,8}$ = Landscape and nature	$U_{1,7}(V_{7,8}) = 0.111$	$U_{2,7}(V_{7,8}) = 0.5$

Table A.47: Travel scenario specification.

Rank	Agent	Score
1	AgentLG	0.642 ±0.011
2-3	CUHKAgent	0.618 ±0.007
2-4	TheNegotiator Reloaded	0.613 ±0.008
3-5	OMACagent	0.601 ±0.009
4-5	IAMhaggler2012	0.600 ±0.006
6	BRAMAgent2	0.570 ±0.013
7	Meta-Agent	0.545 ±0.006
8	AgentMR	0.254 ±0.009

TABLE A.48: Scores in the Travel scenario (averaged over $\delta \in \{0.50, 0.75, 1.00\}, u_{\overline{\alpha}} \in \{0.00, 0.25, 0.50\}$), with 95% confidence intervals.

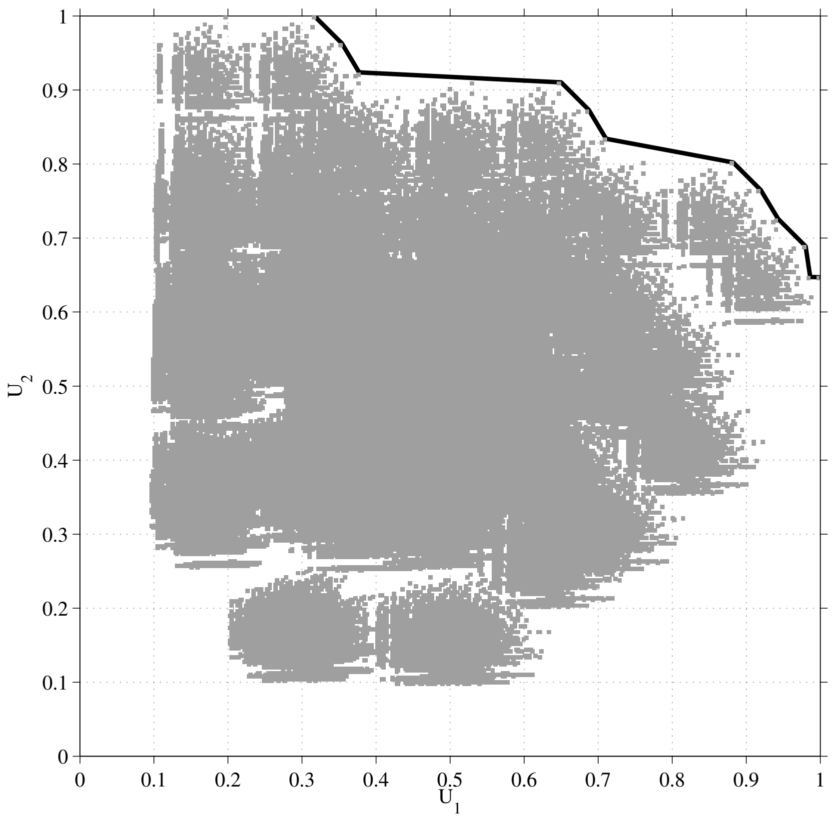


FIGURE A.24: Travel scenario outcome space.

Appendix B

Pareto-Search Selection

In this appendix, we discuss an additional approach for selecting a multi-issue offer with a given utility (other such approaches are described in Section 3.3). This particular approach was designed for use in scenarios which contain issues with and without a known, common ordering. Since this thesis focuses on issues which do not have a known, common ordering, we omit this strategy from the body of the thesis.

The approach, known as Pareto-search, is to search for the offer with a given utility that we consider to be closest to the best offer that we have seen from our opponents. The rationale for this approach is that, if we can select an offer close to the best offer (according to our utility function) seen from the opponent (rather than any of its other offers) it can aid the opponent in learning our preferences. If the opponent uses a similar search method, the agreement that is reached should be Pareto efficient. In what follows, we begin by considering only scenarios where the issue values take a known, common ordering (including continuous issues), before extending the approach to deal with issues where the values do not have a known, common ordering.

B.1 Issues with a known, common ordering

In order to search for an offer, we consider our agent's utility function to be a mapping from a multi-dimensional space (in which there is a dimension representing each issue with a known, common ordering, including continuous issues) to a real value which represents the utility of the outcome. Our strategy treats integer based issues in the same way as continuous issues, except when generating an offer, where it must round the values for each integer based issue, to ensure that their values remain as integers in the offer.

Domain	Profile 1	Profile 2
Issue 1: Price $v_1 \in \mathbb{Z}, \quad 5000 \leq v_1 \leq 15000$	$w_{1,1} = 0.3$ $U_{1,1}(v_1) = -0.0001 \cdot v_1 + 1.5$	$w_{2,1} = 0.8$ $U_{2,1}(v_1) = 0.0001 \cdot v_1 - 0.5$
Issue 2: Batt.Cap. $v_2 \in \mathbb{R}, \quad 5 \leq v_2 \leq 20$	$w_{1,2} = 0.7$ $U_{1,2}(v_2) = \begin{cases} 0.1 \cdot v_2 - 0.5 & \text{if } v_2 \leq 15 \\ -0.2 \cdot v_2 + 4.0 & \text{otherwise} \end{cases}$	$w_{2,2} = 0.2$ $U_{2,2}(v_2) = \begin{cases} 0.2 \cdot v_2 - 1.0 & \text{if } v_2 \leq 10 \\ -0.1 \cdot v_2 + 2.0 & \text{otherwise} \end{cases}$

TABLE B.1: Agent's utilities for the 'Price' and 'Battery Capacity' issues in a modified version of the car sales domain. $w_{p,i}$ is the weight of issue i to agent p and $U_{p,i}(v_i)$ is the utility to agent p of value v_i for issue i.

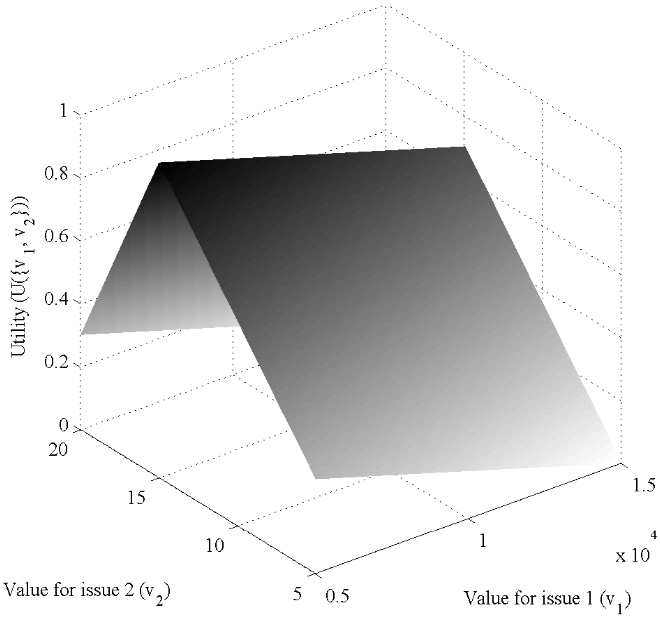


FIGURE B.1: Multi-dimensional space representing the utility of outcomes in the modified car sales domain, with one continuous issue and one integer issue (as described in Table B.1). Values v_1 and v_2 are the values of the two issues. $U(\{v_1, v_2\})$ is the utility of the offer represented by those values.

For example, for a modified version of the car sales domain (introduced in Chapter 1), which consists of only two issues (one continuous, the other integer), as described in Table B.1, this multi-dimensional utility space (the space of all possible outcomes) is shown graphically in Figure B.1.

Here, the values v_1 and v_2 for the two issues are shown on the two horizontal axes. The vertical axis represents the utility $U(\{v_1, v_2\})$ of each possible offer.

Now, by taking a cross-section of the utility space, we can construct an iso-utility space, which is a multi-dimensional space, with the number of dimensions equal to the number of continuous or integer issues. This space consists of outcomes which result in the same utility for our agent. To this end, Figure B.2 shows a number of iso-utility spaces in the

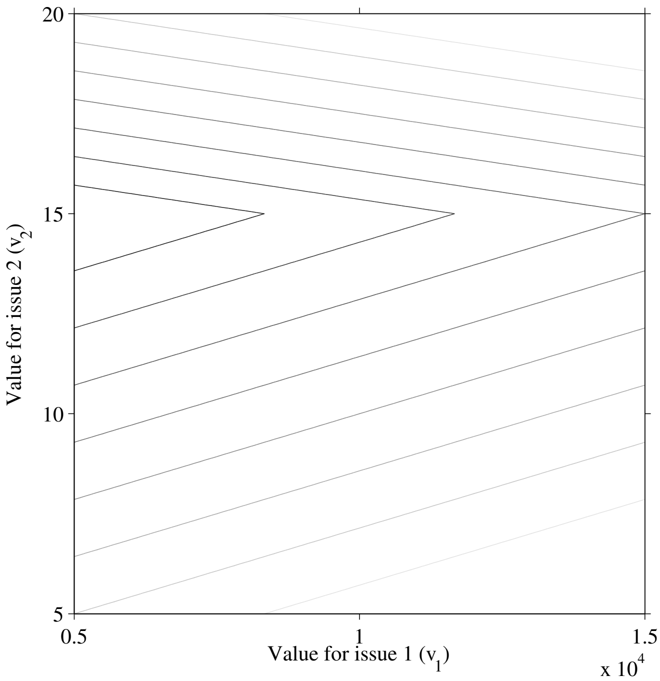


FIGURE B.2: Iso-utility contours in a domain with one continuous issue and one integer issue (as described in Table B.1). Values v_1 and v_2 are the values of the two issues. The darker contours represent the higher utility values.

modified car sales domain. Each iso-utility space is shown as a contour line (the contour lines are displayed at utility intervals of 0.1).

Moreover, the iso-utility space that is chosen at a particular time is the one which represents our current desired utility level (as decided by our concession strategy, which we detailed in Section 3.2). Furthermore, for a given desired utility level (in the following example we choose a utility of 0.8), we can view the iso-utility space as in Figure B.3. Based on the work of Somefun et al. (described in Section 2.3.2.3), we use projection to find the point on our iso-utility space which is closest to the best offer received from our opponent. We extend their work to deal with issues in which there is not a known, common ordering, in Section B.2.

Specifically, in terms of closeness between two offers $\{v_1, v_2, ..., v_n\}$ and $\{v'_1, v'_2, ..., v'_n\}$, we use the Euclidean distance, that is:

$$\sqrt{\sum_{i=1}^{n} \left(\frac{v_i - v'_i}{\text{range}_i} \right)^2} \tag{B.1}$$

where range_i is the range of values allowed for issue i. The reason that we divide by the range is to ensure that the scale of the issue's values does not affect the distance measurement. By performing this division, the value of each issue is normalised by its range, to give it the same weight as all other issues. If this division was not performed,

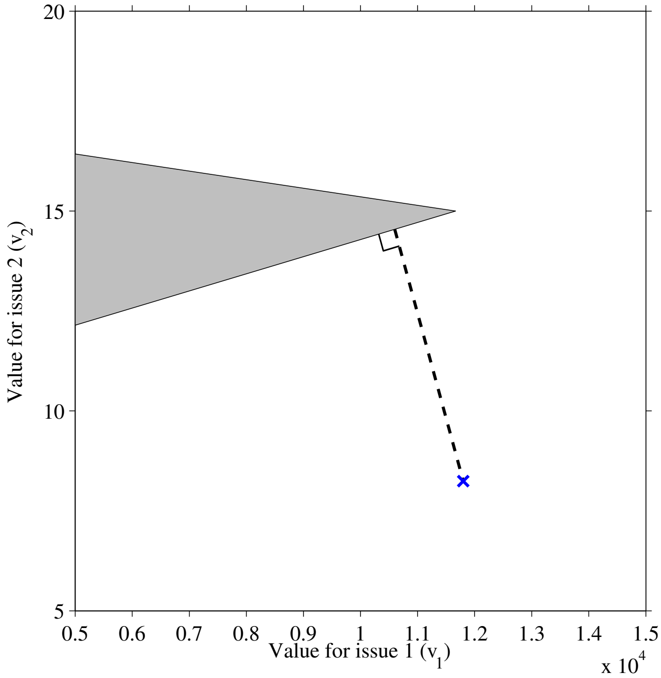


FIGURE B.3: Projection of a point representing an opponent's offer ($\{11800, 8.24\}$, marked with a cross) onto an iso-utility space (at a utility of 0.8) for a domain with two continuous issues (as described in Table B.1). The shaded region represents the space in which the utility is greater than 0.8.

the issue with the larger range would contribute significantly more to the distance calculation. For example, in the modified car sales domain presented in Table B.1, the price issue (with a range of 10000) would have a larger effect than the battery capacity (with a range of 15).

B.2 Issues without a known, common ordering

The approach we have just introduced does not consider domains with discrete issues which do not have a known, common ordering. In order to meet our requirement for a strategy which can negotiate in domains with such issues (Requirement 5), we need to make some modifications to our strategy. In particular, since they do not have a common ordering, we cannot treat them as further dimensions in our space. To address this problem, we first of all continue to create an iso-utility space to represent the issues with a known, common ordering (including continuous issues). However, to handle the additional complexity of discrete issues without a known, common ordering, we create an iso-utility space for each combination of the values of the discrete issues without a known, common ordering. As an example, consider the domain we used earlier, but add a discrete issue *Colour*, as described in Table B.2 (we also reduce the weights of the *Price* issue by 0.1, that is $w_{1,1} = 0.2$ and $w_{2,1} = 0.8$).

Domain	Profile 1	Profile 2
Issue 3: Colour	$w_{1,3} = 0.1$	$w_{2,3} = 0.1$
$V_3 = \{ \quad V_{3,\text{Red}} = \ 1$	$U_{1,3}(V_{3,\text{Red}}) = 3$	$U_{2,3}(V_{3,\text{Red}}) = ,$
$V_{3,\text{Green}} = \ 2$	$U_{1,3}(V_{3,\text{Green}}) = 2$	$U_{2,3}(V_{3,\text{Green}}) = ,$
$V_{3,\text{Blue}} = \ 3$	$U_{1,3}(V_{3,\text{Blue}}) = 1$	$U_{2,3}(V_{3,\text{Blue}}) = \}$

TABLE B.2: Agent's utilities for the 'Colour' issue.

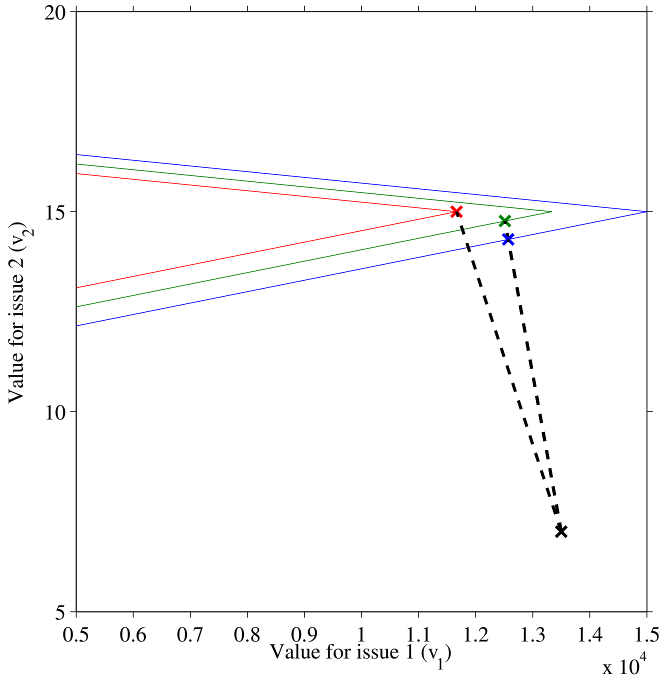


FIGURE B.4: Projection of a point representing an opponent's offer (at $\{13500, 7, v_3\}$, marked with a cross) onto an iso-utility space (at a utility of 0.8) for a domain with two continuous issues and a discrete issue (as described in Tables B.1 and B.2). The projections are shown for each of the three combinations of values for the issues without a common ordering.

For each combination of the values for the issues without a common ordering, we create a multi-dimensional space, and use the iso-utility projection method (described in Section 3.3) to find a solution (where possible) for each of these combinations. We demonstrate this projection in Figure B.4, where the opponent's offer $\{13500, 7, v_3\}$ (we write v_3 to represent the value of issue 3, since it does not affect the projection) is projected to give the solutions $\{11667, 15.0000, \text{Red}\}$, $\{12514, 14.7659, \text{Green}\}$ and $\{12572, 14.3063, \text{Blue}\}$. For some combinations, the maximum overall utility available from the package with the best values for the other issues may be lower than our current utility level. In this case, there will not be a solution which contains this combination of values for the issues without a common ordering.

Once a solution has been found for each combination of the values, it is necessary for our agent to choose one of these solutions as its offer. Our agent is indifferent between each of these solutions, as they all belong to the same iso-utility space, resulting in them having an identical utility. However, their utility to the opponent may vary, and in order to negotiate efficiently, we should choose the one which offers the highest utility to the opponent.

Now, since we assume that the opponent's utility function is unknown (Requirement 3), we need a way to estimate it. This can be done using the approach taken by Hindriks and Tykhonov (2008) and as described in Section 3.3, using Bayesian updating to learn the preferences of the opponent. The agent then evaluates the solution for each combination of values using our model of the opponent's utility function, in order to obtain an estimate of the utility of the offer to the opponent. The overall solution that is chosen is then the one which maximises the opponent's utility according to our model of its utility function.

In order to ensure that this approach remains computationally tractable (Requirement 7) even in domains with large outcome spaces, it may be necessary to limit the number of combinations of values that we perform the iso-utility projection method for. A solution to this is to choose a number of combinations (several hundred), by identifying those which maximise the sum of our utility and our opponent's utility (according to our model of the opponent's utility function).

References

S. Aknine, S. Pinson, and M. F. Shakun. An extended multi-agent negotiation protocol. *Autonomous Agents and Multi-Agent Systems*, 8:5–45, 2004.

B. An. *Automated Negotiation for Complex Multi-Agent Resource Allocation*. PhD thesis, University of Massachusetts - Amherst, 2011.

B. An, N. Gatti, and V. R. Lesser. Extending alternating-offers bargaining in one-to-many and many-to-many settings. *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 423–426, 2009.

B. An, V. R. Lesser, and K. M. Sim. Decommitment in multi-resource negotiation. *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems*, 3:1553–1556, 2008.

B. An, V. R. Lesser, and K. M. Sim. Strategic agents for multi-resource negotiation. *Autonomous Agents and Multi-Agent Systems*, 2011.

M. R. Andersson and T. Sandholm. Leveled commitment contracts with myopic and strategic agents. *Journal of Economic Dynamics and Control*, 25:615–640, 2001.

T. Baarslag, K. Hindriks, C. M. Jonker, S. Kraus, and R. Lin. The first automated negotiating agents competition (ANAC 2010). *New Trends in Agent-based Complex Automated Negotiations; Series of Studies in Computational Intelligence*, 2010.

M. Bichler. Trading financial derivatives on the web—an approach towards automating negotiations on otc markets. *Information Systems Frontiers*, 1(4):401–414, 2000.

K. Binmore. *Fun and Games*. D. C. Heath and Company, 1992.

T. Bosse and C. M. Jonker. Human vs. computer behaviour in multi-issue negotiation. *Proceedings of the First International Workshop on Rational; Robust; and Secure Negotiations in Multi-Agent Systems*, pages 11–24, 2005.

F. Brandt, T. Sandholm, and Y. Shoham. Spiteful bidding in sealed-bid auctions. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 1207–1214, 2007.

F. Brazier, F. Cornelissen, R. Gustavsson, C. M. Jonker, O. Lindeberg, B. Polak, and J. Treur. Agents negotiating for load balancing of electricity use. *Proceedings of the 18th International Conference on Distributed Computing Systems*, pages 622–629, 1998.

R. M. Coehoorn and N. R. Jennings. Learning on opponent's preferences to make effective multi-issue negotiation trade-offs. *Proceedings of the Sixth International Conference on Electronic Commerce*, pages 59–68, 2004.

J. Dang and M. N. Huhns. An extended protocol for multiple-issue concurrent negotiation. *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 1: 65–70, 2005.

P. Faratin, C. Sierra, and N. R. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24:159–182, 1998.

P. Faratin, C. Sierra, and N. R. Jennings. Using similarity criteria to make issue tradeoffs in automated negotiations. *Artificial Intelligence*, 142:205–237, 2002.

S. Fatima, M. Wooldridge, and N. R. Jennings. Optimal negotiation strategies for agents with incomplete information. *Agent Theories; Architectures; and Languages*, pages 53–68, 2001.

S. Fatima, M. Wooldridge, and N. R. Jennings. Multi-issue negotiation under time constraints. *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 143–150, 2002.

S. Fatima, M. Wooldridge, and N. R. Jennings. Optimal negotiation of multiple issues in incomplete information settings. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 3:1080–1087, 2004.

S. Fatima, M. Wooldridge, and N. R. Jennings. Multi-issue negotiation with deadlines. *Journal of Artificial Intelligence Research*, 27:381–417, 2006.

S. Fatima, M. Wooldridge, and N. R. Jennings. On efficient procedures for multi-issue negotiation. *Lecture Notes in Computer Science*, 4452:31–45, 2007.

K. Fischer, J. P. Müller, and M. Pischel. Cooperative transportation scheduling: an application domain for dai. *Journal of Applied Artificial Intelligence*, 10:1–33, 1995.

J. C. Harsanyi and R. Selten. A generalized Nash solution for two-person bargaining games with incomplete information. *Management Science*, 18:80–106, 1972.

K. Hindriks, C. M. Jonker, S. Kraus, R. Lin, and D. Tykhonov. GENIUS: negotiation environment for heterogeneous agents. *Proceedings of the Eighth International Joint Conference on Autonomous Agents and Multiagent Systems*, 2:1397–1398, 2009a.

K. Hindriks, C. M. Jonker, and D. Tykhonov. The benefits of opponent models in negotiation. *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2:439–444, 2009b.

K. Hindriks and D. Tykhonov. Opponent modelling in automated multi-issue negotiation using bayesian learning. *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems*, 1:331–338, 2008.

HSL. A collection of fortran codes for large scale scientific computation, 2011. `http://www.hsl.rl.ac.uk`.

N. R. Jennings, P. Faratin, A. R. Lomuscio, S. D. Parsons, M. Wooldridge, and C. Sierra. Automated negotiation: prospects; methods and challenges. *Group Decision and Negotiation*, 10:199–215, 2001.

P. R. Jordan, C. Kiekintveld, and M. P. Wellman. Empirical game-theoretic analysis of the tac supply chain game. *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1188–1195, 2007.

E. Kalai. Proportional solutions to bargaining situations: interpersonal utility comparisons. *Econometrica: Journal of the Econometric Society*, 45:1623–1630, 1977.

E. Kalai and M. Smorodinsky. Other solutions to Nash's bargaining problem. *Econometrica: Journal of the Econometric Society*, 43:513–518, 1975.

G. Lai, K. Sycara, and C. Li. A decentralized model for multi-attribute negotiations. *Proceedings of the Eighth International Conference on Electronic Commerce*, 156:3–10, 2006.

R. Lin and S. Kraus. Can automated agents proficiently negotiate with humans? *Communications of the ACM*, 53:78–88, 2010.

R. Lin, S. Kraus, J. Wilkenfeld, and J. Barry. An automated agent for bilateral negotiation with bounded rational agents with incomplete information. *Frontiers in Artificial Ingelligence and Applications*, 141:270–274, 2006.

Z. A. Livne. *The role of time in negotiations.* PhD thesis, Massachusetts Institute of Technology, 1979.

R. B. Myerson. Utilitarianism; egalitarianism; and the timing effect in social choice problems. *Econometrica: Journal of the Econometric Society*, 49:883–897, 1981.

J. Nash. The bargaining problem. *Econometrica: Journal of the Econometric Society*, 18:155–162, 1950a.

J. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the United States of America*, 36:48–49, 1950b.

J. Nash. Two-person cooperative games. *Econometrica: Journal of the Econometric Society*, 21:128–140, 1953.

T. D. Nguyen and N. R. Jennings. Concurrent bi-lateral negotiation in agent systems. *Fourteenth International Workshop on Database and Expert Systems Applications*, pages 844–849, 2003.

T. D. Nguyen and N. R. Jennings. Managing commitments in multiple concurrent negotiations. *Electronic Commerce Research and Applications*, 4:362–376, 2005.

M. J. Osborne and A. Rubinstein. *Bargaining and Markets.* Academic Press, 1990.

S. D. Parsons and N. R. Jennings. Negotiation through argumentation a preliminary report. *Proceedings of the Second International Conference on Multiagent Systems*, pages 267–274, 1996.

A. Pommeranz, J. D. Broekens, W. Visser, W.-P. Brinkman, P. Wiggers, and C. M. Jonker. Multi-angle view on preference elicitation for negotiation support systems. *Proceedings of First International Working Conference on Human Factors and Computational Models in Negotiation*, pages 19–26, 2008.

I. Ponka. *Commitment models and concurrent bilateral negotiation strategies in dynamic service markets.* PhD thesis, University of Southampton, 2009.

H. Raiffa. *The Art and Science of Negotiation.* Harvard University Press, 1982.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, 2006.

D. M. Reeves. *Generating Trading Agent Strategies: Analytic and Empirical Methods for Infinite and Large Games.* PhD thesis, University of Michigan, 2005.

J. S. Rosenschein and G. Zlotkin. *Rules of Encounter.* The MIT Press, 1994.

A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, 50:97–109, 1982.

A. Rubinstein. *Modeling bounded rationality.* The MIT Press, 1998.

S. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach.* Pearson Education, 2003.

T. Sandholm and V. R. Lesser. Leveled commitment contracts and strategic breach. *Games and Economic Behavior*, pages 212–270, 2001.

T. Sandholm and N. Vulkan. Bargaining with deadlines. *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1:44–51, 1999.

B. Shi and K. M. Sim. A regression-based coordination for concurrent negotiation. pages 699–703, 2008.

B. Shi, E. H. Gerding, P. Vytelingum, and N. R. Jennings. An equilibrium analysis of market selection strategies and fee strategies in competing double auction marketplaces. *Autonomous Agents and Multi-Agent Systems*, 2012.

R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29:1104–1113, 1980.

L.-K. Soh and C. Tsatsoulis. A real-time negotiation model and a multi-agent sensor network implementation. *Autonomous Agents and Multi-Agent Systems*, 11:215–271, 2005.

D. J. A. Somefun, E. H. Gerding, and J. A. L. Poutré. Efficient methods for automated multi-issue negotiation: Negotiating over a two-part tariff. *International Journal of Intelligent Systems*, 21:99–119, 2006.

Student. The probable error of a mean. *Biometrika*, 6:1–25, 1908.

P. J. 't Hoen and J. A. L. Poutré. A decommitment strategy in a competitive multi-agent transportation setting. *Agent-Mediated Electronic Commerce V*, 3048:56–72, 2004.

A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.

E. Weisstein. *CRC Concise Encyclopedia of Mathematics.* CRC Press, 1999.

B. L. Welch. The generalization of 'student's' problem when several different population variances are involved. *Biometrika*, 34:28–35, 1947.

M. P. Wellman. Methods for empirical game-theoretic analysis. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 2:1552–1555, 2006.

M. Wooldridge. *An Introduction to Multiagent Systems.* John Wiley Sons, 2009.

M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10:115–152, 1995.

Y. Yasumura, T. Kamiryo, S. Yoshikawa, and K. Uehara. Acquisition of a concession strategy in multi-issue negotiation. *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 7:161–171, 2009.

S. Yoshikawa, Y. Yasumura, and K. Uehara. Strategy acquisition on multi-issue negotiation without estimating opponent's preference. *Lecture Notes in Computer Science*, 4953:371–380, 2008.

D. Zeng and K. Sycara. Bayesian learning in negotiation. *International Journal of Human-Computers Studies*, 48:125–141, 1998.

F. Zeuthen. *Problems of monopoly and economic warfare.* George Routledge & Sons, 1930.