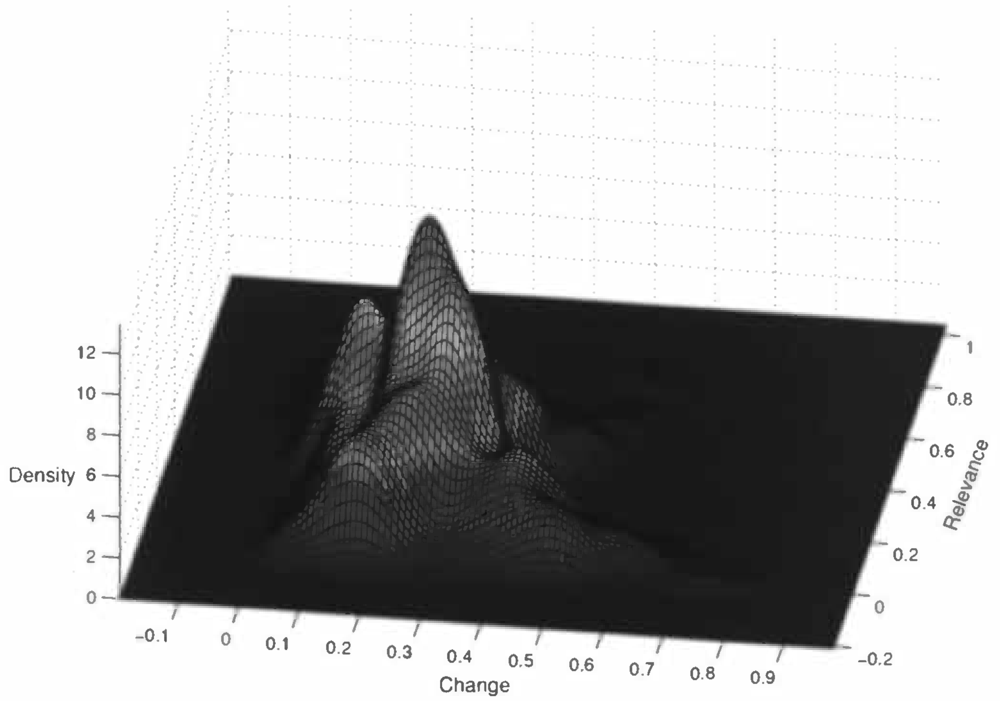
955
2004
014

University of Groningen

LEARNING AN OPPONENT'S PREFERENCES

In Order to Make Effective Multi-Issue Negotiation Trade-Offs

Robert Martijn Coehoorn



Internal Supervisor:
Dr. Rineke Verbrugge
University of Groningen
The Netherlands

External Supervisor
Prof. Nicholas R. Jennings
University of Southampton
United Kingdom

968.

University of Groningen

LEARNING AN OPPONENT'S PREFERENCES IN ORDER TO MAKE EFFECTIVE MULTI-ISSUE NEGOTIATION TRADE-OFFS

by
Robert Martijn COEHOORN

A thesis submitted in the partial fulfillment for the degree of Master of Science

in the
Faculty of Psychology, Pedagogical and Sociological Sciences
Department of Artificial Intelligence

June, 2004

Internal Supervisor:
Dr. Rineke Verbrugge
University of Groningen
The Netherlands

External Supervisor
Prof. Nicholas R. Jennings
University of Southampton
United Kingdom

Abstract

Software agents that autonomously act and interact to achieve their design objectives are increasingly being developed for a range of electronic commerce applications (i.e. commercial activity conducted via electronic media). Also, within an agent-oriented view of computation, it is readily apparent that most problems require or involve multiple agents which have to interact. In this context, automated negotiation is a central concern since it is the de facto means of establishing contracts for goods or services between agents.

In many of these cases, these contracts consist of multiple issues (e.g. price, time of delivery, quantity, quality) which makes the negotiation more complex than when dealing with just price. In particular, effective and efficient multi-issue negotiation requires an agent to have some indication of its opponent's preferences over these issues.

However, in competitive domains, such as e-commerce, an agent will not reveal this information and so the best that can be achieved is to learn some approximation of it through the negotiation exchanges. To this end, the use of a statistical method, *kernel density estimation*, was explored and evaluated. Specifically, the work is couched in the context of making negotiation trade-offs: Giving in on one issue, while simultaneously demanding more on another. This approach proved to make the negotiation outcome more efficient for both participants.

Contents

Acknowledgements

Before starting, I would like to thank some people for their support throughout this thesis. The whole experience, thanks to you all, greatly enriched my academic and social life.

First, a warm thanks goes out to my external supervisor Nick Jennings. After accepting me at a great group, he supported me amazingly by spending a lot of time, energy and enthusiasm in my research. Furthermore, he kept me motivated ("research is not a linear activity") and critical by asking the right question at the right time, forcing me to go that one step further. Back in the Netherlands, the enthusiasm of Rineke Verbrugge, my internal supervisor, kept me going through that important final stages, and her broad knowledge of the domain were of great help while writing my thesis.

I would also like to acknowledge the help of all the people at the IAM-group in Southampton. The kindness and expertise within this group often proved of great help. Specific thanks to Steve Gunn, for bringing kernel density estimation to my attention. But also Minghua He, Raj Dash, Etty David, Partha Dutta, Paul Groth, Perukrishnen Vytelingum, Maria Karam, Chris Bailey, Cora Excelente-Toledo, Shaheen Fatima, and all the others at IAM who helped me whenever possible and motivated me to get the best out of myself. And back at the university of Groningen, the conversations, coffee-breaks and ping-pong matches with, among others, Berto Booijink, Hein van der Ploeg and Jan Willem Hiddink were invaluable when the end came near.

Last, but not least, special thanks to my friends and family in the Netherlands and the UK, who kept supporting me throughout this period. All those lines from the Netherlands were warmly appreciated, as well as all the socials, friendship and attention in England. A special thanks goes to Irene, who supported me so often when I thought I would never get where I am now.

All of you, also those I didn't explicitly mention, thanks.
Bedankt!

Chapter 1

Introduction

Automated negotiation is becoming more and more important in several fields of artificial intelligence and business[1]. In multi-agent systems for example, where inter-dependencies between the agents are inevitable, it is a means for agents to communicate and compromise to reach (mutually beneficial) agreements. In such situations, the agents have a common interest in cooperating, but conflicting interests on how to cooperate. Examples of such systems include intelligent agents that negotiate over the meeting schedule of people for whom they work (Bui et al., 1996), bargaining systems that try to negotiate the best deal (Zeng and Sycara, 1998) or (simulated) robotic system that have to complete a common objective (Rosenschein and Zlotkin, 1994).

On the other hand there is electronic commerce (e-commerce), which is increasingly assumed to play a pivotal role in many organizations. "It offers opportunities to significantly improve (make faster, cheaper, more personalized, an/or more agile) the way that businesses interact with both their customers and their suppliers" (He et al., 2003).[2] Within this domain, automated negotiation is the de facto means of establishing contracts for goods or services between agents. Examples within this domain are for example the *trading agent competition* (TAC), in which "travel agents" have to assemble travel packages on behalf or their user (He and Jennings, 2003)[3] or models for agent-based supply chain management, complex systems of business units that convert raw materials to consumer products.

To this end, software agents that autonomously act and interact to achieve their design objectives are increasingly being developed for a range of applications (see He et al. (2003) for a review of the applications in e-commerce). In such agent-mediated applications, a key component of the solution is the way in which the agents negotiate to establish contracts with one another to provide particular services or goods under particular terms and conditions.

In many cases, it is important that the agents do not only bargain over the price of a good, but also take into account aspects like delivery time, quality, and payment method. We will call these multi-dimensional goods *services*. This

[1]see (Weiss, 1999, ch. 13) for an overview of applications.

[2]In this thesis, many of the examples and applications will be from this latter domain. However, it should be stressed that the application of the method proposed here is not limited to this domain.

[3]see `http://www.sics.se/tac/` for details of the competition.

point of view on goods is in line with the more economical *consumer theory*, in which goods are viewed as consisting of properties from which utility for the consumer is derived, as opposed to the approach that goods are the direct object of utility (Lancaster, 1966).

Different parties will typically assign different relative importance to these attributes (issues) of a service. Therefore, in negotiation over such *multi-issue* contracts it is often possible to reach an agreement that is mutually beneficial for both parties Raiffa (1982). For example, when buying a flight-ticket, the time of day of the flight might not be of great importance to the buyer, whereas the travel agency selling the ticket might think this issue very important because of the number of seats available in the airplane. This difference in importance (weight) attached to the different issues by the different agents provides the opportunity of joint improvement in negotiations over such a service. However, an impediment to this win-win scenario occurs in many areas of application, such as e-commerce, because the agents are unlikely to truthfully reveal their preferences, utility functions or reservations values (i.e. the maximum (or minimum) value acceptable for the agent) for fear of being exploited. In such circumstances, the best that can be achieved is to try and approximate these preferences based upon past experience and the offers and counter-offers that the opponent makes during the current negotiation encounter.

Against this background, the aim of this thesis is to report on the development of a novel method for attempting to learn the negotiation preferences of the opponent. First, however, we define exactly what we mean by the terms "agent" and "electronic commerce" and lay out the requirements of our research, as well as the method we propose.

1.1 Agents

An increasing number of computer systems are being viewed as *autonomous agents*. The reasons for this are twofold. First, agents are advocated as the next model for engineering complex, distributed systems (e.g. Wooldridge (1997)). Secondly, agents could provide artificial intelligence with an overarching framework, bringing the components of the different subdisciplines together to one intelligent entity (Russell and Norvig, 2003). Although there is not one single definition what an agent is, software that is considered to be an agent should at least exhibit some properties (Wooldridge, 1997):

- It needs to be *autonomous*: capable of execution without direct intervention (human or other) and have control over their own state.
- It needs to be *reactive*: able to respond rational to events that occur by maintaining an ongoing interaction with their environment.
- It needs to be *proactive*: besides simply responding to their environment, they act to achieve their goals and by this means meet their owner's objectives.

Thus, for example, in the e-commerce setting an agent might be instructed to find a reasonably priced digital camera with some constraints on the technical specifications, and that has to be delivered within three weeks. Therefore, it should be autonomous since it should look for the product without intervention of the user, it should be reactive to cope with a dynamic environment (e.g. the internet) and it should be proactive by fulfilling its goal of finding such a product when it is available.

1.2 Electronic Commerce

Electronic commerce, or *e-commerce* is defined by the Oxford English Dictionary[4] as:

> Commercial activity conducted via electronic media, especially on the Internet; the sector of the economy engaged in such activity.

The field of electronic commerce is most applied to business-to-business (B2B) and business-to-customer (B2C), where the former refers to transactions where both the seller and the buyer are business corporations and the latter case refer to shoppers buying products in an on-line store. Especially B2B business is expected to be the predominant means of e-commerce, with an expected revenue in the range of 800 billion dollar in 2004 (as opposed to 180 billion dollars for B2C)[5]. By the means of electronic commerce, many commercially interesting activities within this B2B activity, such as dynamic pricing, the ability to easily compare many goods, and the ability to negotiate contracts much more frequently are made possible (Rosenschein and Zlotkin, 1994). To achieve such systems, agents will be more and more interacting entities, and negotiation is the de facto means of establishing contracts with other agents for goods or services.

1.3 Requirements

The field of automated negotiation is vast and very inter-disciplinary. To get a better focus within this field, the requirements of this work are laid out in this section. The source of these requirements is twofold. First, the domain of the problem as a competitive, multi-issue negotiations implies that the opponent will try to get the best possible deal. Thus, it is possible the opponent changes its strategy over time, requiring an adaptive system. This is extended by the dynamic environment in which the agent will operate. Furthermore, since the knowledge about the opponent will be limited the solution should not make rigid assumptions or require an extensive knowledge of the opponent. Second, a practical solution is sought. This implies limited computational power and constraints on the sources. Also, a theoretical solution which can not be applied to real scenarios does not suffice.

Following this line of reasoning, the requirements can be stated as follows:

i) The agents will be bounded in their computational power. While this may not always lead to the optimal solution, it will provide a solution which is useful in practice.

[4]Oxford English Dictionary, Draft entry September 2001, Oxford University Press (`http://www.oed.com`)

[5]according to eMarketer: `http://www.emarketer.com`.

ii) The domain will be bounded in resources. This means in our experiments most importantly that an agreement should be made within a reasonable amount of time. However, it should also be possible to limit the use of other resources.

iii) We will focus on the process of the negotiation, as opposed to merely the best solution. This is necessary to be applicable to real-world problems.

iv) The contracts are assumed to consist of multiple issues, since in most domains this will be the case.

v) The solution should be as generic as possible. Thus, the implementation of the method presented should be possible without an extent knowledge within the domain of negotiation or game theory.

vi) The method should be adaptive to the environment or opponent.

Using these requirements, an evaluation of the related work within automated negotiation will be made possible, and a basis for a structural analysis is provided.

1.4 Method

To make a further progression in the field of automated negotiation, this thesis reports on the development of a novel method for attempting to learn the negotiation preferences of the opponent. Specifically, we try to learn this information with respect to the provision of a particular service (since an agent's preferences may vary for different services). The particular approach we use is *kernel density estimation* (*KDE*) which is a statistical method known to provide a simple way of finding structure in data sets without the imposition of a parametric model (Wand and Jones, 1995). It works in the following way; any data that is available about previous negotiation encounters for the provision of a particular service is processed offline (as described in section 3.2) to acquire a probability density function over the opponent's likely weights for the various issues. This function can then be augmented by online learning that reflects new information emerging from the ongoing encounter.

The KDE-method was chosen for two main reasons. First, following the requirements in section 1.3, the computational complexity of the model is important because agents are bounded in their computational power. It is shown in section 3.2 that learning a KDE can be done in $n \log n$ complexity. When it is learned, the lookup of a prediction is constant. Thus, the method can indeed be applied in computationally bounded agents.

Second, we want to use the method for a wide variety of opponents with varying strategies without having to fundamentally alter the model. Hereto, we want to make as few explicit assumptions about the relation between the negotiation history and the importance of the negotiation issue as possible. One of the properties of KDE is that it is a *non-parametric* method, meaning that no assumptions about the underlying function has to be made; for example that the relation sought for is linear or gaussian. More specifically, the relation between time, negotiation history and issue-weight does not have to be specified a priori. In contrast, when using parametric regression methods such as linear regression or the expectation-maximization algorithm an assumption must be made about the underlying distribution function (respectively linear and Gaussian). Similarly, in Bayesian learning, an often used paradigm within automated negotiation, an a priori distribution must be given for the weights of the opponent (see section 2.4.3 for more detail), whereas in KDE the initial distribution is based solely on the training data. This, the method complies with the requirement that the solution should be generic.

We choose to evaluate the efficacy of KDE in the context of making negotiation *trade-offs* in bilateral encounters. We focus on trade-offs because they are a key feature of bargaining behaviour and cannot be achieved without a reasonable degree of information about an opponent. In making a trade-off an agent concedes on one issue and demands more on another. Overall, the aim is for the agent to keep a constant utility compared to its previous offer for itself, but to increase the utility of the opponent. Hence this makes the trade-off more likely to be accepted by the opponent. For the actual computation of the trade-offs, we use Faratin's algorithm (Faratin et al., 2002) which evaluates possible trade-offs based on fuzzy similarity. Thus while this algorithm makes use of the fact that different negotiation issues are of different degrees of importance to the agent, it does not actually provide a method for learning these weights.

Against this background, this research advances the state of the art in the following ways. First, KDE has never been used to learn the preferences of a negotiation opponent. Moreover, although we demonstrate it in the context of making negotiation trade-offs, the method can be applied to other aspects of negotiation where more accurate knowledge about an opponent's preferences can lead to better negotiation outcomes. Second, we extend Faratin's trade-off algorithm by incorporating a learning model, thereby making it more effective in finding trade-offs in a wider variety of circumstances.

1.5 Structure of Thesis

The remainder of the thesis is structured as follows. Chapter 2 discusses related work in the area of negotiation and learning in negotiation. Several methods for coordination and negotiation models are introduced and critically evaluated against the requirements of our research. After that, we describe the theoretical models that underpin this work; namely, the negotiation model, the KDE-method and the trade-off algorithm (chapter 3). The empirical evaluation of the developed model is presented in chapter 4. Finally, in chapter 5.1 we present the conclusions and outline avenues for future work.

Chapter 2

Related Work

To place this research in the history of Artificial Intelligence (AI), we have to start looking at distributed artificial intelligence (DAI), which evolved and diversified rapidly since the mid to late 1970s. Due to the interdisciplinary nature of the field, including AI, computer science, sociology, economics, organization and management science, and philosophy, a precise definition is hard to get. Here, the definition of Weiss (1999) is used:

> DAI is the study, construction, and application of multiagent systems, that is, systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks.

An agent is here defined as a computational entity (e.g. a software program or a robot) that can be viewed as perceiving and acting upon its environment and that is autonomous in that its behavior at least partially depends on its own experience (*ibid.*). This field of research later resulted in other forms of interactions, with perhaps negotiation as the most fundamental one for cooperative as well as competitive domains:

> [...] perhaps the most fundamental and powerful mechanism for managing inter-agent dependencies at run-time is *negotiation* [...]. Negotiation underpins attempts to cooperate and coordinate (both between artificial and human agents) and is required both when the agents are self-interested and when the are cooperative. It is so central precisely because the agents are autonomous. (Jennings et al., 2001)

For a more specific definition, Walton and Krabbe (1995) look at negotiation using the approach of formal dialogue games from a more philosophical stance. They focus on competitive domains, stating that negotiation is self-interested bargaining: "Each participant aims to maximise his share of some goods or services which are in short supply." (Walton and Krabbe, 1995, p. 72). This implies the aim for the agent is to get the best deal out of the negotiation for itself. However, besides this strictly competitive stance, the agents do have the need to cooperate, and therefore the deal should be mutually acceptable.

Against this background, this work focuses on bilateral encounters (encounters between two agents) in which both parties want to reach a mutually acceptable agreement on some matter. The means of achieving this state are to make proposals, trade options and offer concessions.

Given its ubiquity and importance within many different contexts, many different approaches (e.g. from game theory, AI, and social psychology) have been used in negotiation theory. When looking in more detail at this research, however, automated negotiation can be considered to deal with three broad topics according to Jennings et al. (2001):

- *Negotiation protocols*: The set of rules that govern the interaction. These cover for example the number of participants, the level of knowledge of these participants and the valid actions in particular states.
- *Negotiation contracts*: The range of issues over which agreement must be reached. This can be at the one extreme just one issue (such as price), while on the other extreme this can cover hundreds of issues (price, quality, quantity, delivery time, penalties, etc.).
- *Reasoning models*: Covers the decision making model which is employed by the participant to act in line with the protocol in order to achieve their objective.

In this section relevant work in the areas of automated negotiation (in general) and learning in negotiation (in particular) is reviewed. We will discuss the most relevant works within the above given topics against the requirements of this work (respectively section 2.2, 2.3 and 2.4). First, however, section 2.1 looks at some basic terminology of *game theory*, a branch of economics which studies interactions between self-interested agents, and which therefore provides a very useful tool to evaluate the results of automated negotiation.

2.1 Game Theoretic Approach to Bargaining

The central focus of economic models is to find an optimal allocation of scarce resources through cooperation and coordination mechanisms and bargaining (Binmore and Dasgupta, 1989). Thus, how to achieve a common objective by working together and how to arrange actions to be performed in a coherent manner. As in the definition by Jennings given above, the attempt to cooperate and coordinate is stressed. On the other hand, an optimal allocation indicates an allotment of the issues which is acceptable for both agents and as good as possible for the individual agents, stressing the fact that agents are self-interested, but a need for cooperation exists (as implied by the above mentioned definition of (Walton and Krabbe, 1995)). Therefore, the close relation between this field and automated negotiation makes a study of these models worthwhile.

More specifically, a subclass of these economical models are the micro-economic models derived from *game theory*[1]. In these models, which have their origins in the seminal work of von Neumann and Morgenstern (1944), the encounters between agents are viewed as games. In fact, every interaction between two agents is considered to be a game when the agents are conscious of the fact

[1]The standard game theory terms are explained in any of the classic textbooks, such as (Binmore, 1992). A more micro-economical book is (Mas-Colell et al., 1995). Both were used extensively during the writing of this section.

that their actions affect each other (Rasmussen, 1989). For example, when driving a car in a busy street, you play a game with the drivers of the other cars. When a company wants to decide whether it should enter a market or not, it is playing a game with the current incumbents. Also, a buyer is playing a game when negotiating the price of a digital camera on the internet.

This section will discuss the aims and motivations of game theory (2.1.1). Also, a measure of efficiency of the possible deals provided by this field of research is discussed (2.1.2).

2.1.1 Aims and Motivation of Game Theory

To look at the aims and motivations of game theory it is important to give a clear definition first. As said before, game theory analyses what happens in games between agents. We therefore define a game to consist of its players, and a set of rules of the game. The players of the game are described by their preferences, represented by a payoff or utility function defined on the set of possible outcomes[2], and by their beliefs, formally represented by a probability distribution function over a set of possible states of the world. A player then formulates a priori a *strategy* given the rules.

The rules define what a player can do, and when he can do this. It also contains the information available to the agent at each possible decision point. Finally, it defines the payoffs for the players when the game is over.

For example, consider the following situation. Suppose a house is for sale. It is worth £2m for its owner, and £3m for the potential buyer. Hence a deal is possible. When the parties come together, the only thing they have to do is to divide the "surplus" of £1m. An abstraction of this problem leads to a standard bargaining game known as "dividing the dollar" (see e.g. Binmore (1992)): suppose A and B have to divide a dollar. If they reach an agreement on this matter, they each get the agreed amount. If they don't come to an agreement, each gets nothing. Formally, this game can be defined as follows:

Example 1 *Dividing the dollar*

Players:	There are two players A and B.
Rules:	The players have to decide how to divide the dollar by a sequence of offers and counter-offers. Player A bids first. Both player observe all choices previously made. When the players come to an agreement they each get the agreed amount. When no agreement is made they don't get anything.
Outcome:	The outcome space of pairs $m = (m_A, m_B)$, where m_A, m_B is the amount player A respectively B gets, is defined as the set $M = \{m : m_A + m_B \leq 1\}$.
Payoff:	The payoff function of both players is assumed to be equal to the amount of money they get.

■

[2]Note that this payoff function may be, as Rasmussen (1989) points out carefully: i) the payoff the player receives after all players have picked their strategy and the game has been played out; or ii) the expected utility he receives as a function of the strategies chosen by himself and the other players. Although these definitions are distinct, the term "payoff" will be used for both, as is usual in the literature

More formally, a game is described in normal form as follows (Mas-Colell et al., 1995):

Definition 2.1 (Normal form representation) *For a game with I players, the* Normal form representation Γ_N *specifies for each player i a set of strategies S_i (with $s_i \in S_i$) and a payoff function $u_i(s_1, \ldots, s_I)$ giving utility levels associated with the (possibly random) outcome arising from strategies $(s_1, \ldots, s_I)$. Formally we write $\Gamma = [I, \{S_i\}, \{u_i(\cdot)\}]$.*

Given this formal definitions of a game, the aim of game theory is to analyse the players' best choices. This is motivated by the reductionistic nature of an agent: as Binmore (1990) defines, an agent is an optimiser of some function, be it genetic prosperity or maximisation of profit. In the field of game theory, this function is the utility function; a function that maps the preferences of the agent over all possible contracts to a 'score'. In turn, the preferences of an agent are defined by a preference relationship $\preceq$ defined on the set of outcomes Ω. We assume such a preference relationship to be complete and transitive; thus, for all a, b, c:

$$a \preceq b \vee b \preceq a \qquad \text{completeness} \tag{2.1}$$

$$a \preceq b \wedge b \preceq c \Rightarrow a \preceq c \qquad \text{transitivity} \tag{2.2}$$

The former ensures that an agent is always able to express a preference between any two outcomes. The latter is slightly less intuitive, but illustrated by an example of Binmore (1992): Suppose transitivity does not hold. In that case, it is possible an agent holds the following preferences over three outcomes a, b and c:

$$a \preceq b \preceq c \prec a \tag{2.3}$$

Now suppose these outcomes can be traded. The agent likes b at least as much as a, so he should be willing to trade a for b. The same holds for b and c. However, the agent *strictly* prefers a over c, hence he should be willing to pay a small amount of money to trade c for a. After this trade, the agent is in the same state as before the trades, however with less money. This can be repeated ad infinitum, resulting in a money leak.

Given such a preference relationship, the utility function can now be defined formally, where a more preferable contract gets a higher utility. This results in the following definition (Mas-Colell et al., 1995):

Definition 2.2 (Utility function) *A function $u : \Omega \to \mathbb{R}$ is a* utility function *representing the preference relationship $\preceq$ if and only if for all $a, b \in \Omega$:*

$$u(a) \leq u(b) \Leftrightarrow a \preceq b$$

Note that this function representing the preference relationship $\preceq$ is not unique. It is only the ranking of the contracts that matter. To illustrate this, suppose a function $f : \mathbb{R} \to \mathbb{R}$ is a strictly increasing function. Now, if $u(x)$ is a utility function of the preference relation $\preceq$, then $v(x) = f(u(x))$ also represents $\preceq$, This property of a utility function is called *ordinality*. By applying such transformations it is possible to define a utility function that maps the preference relation to the domain [0,1]. Throughout this work, such a transformation will be applied for simplicity.

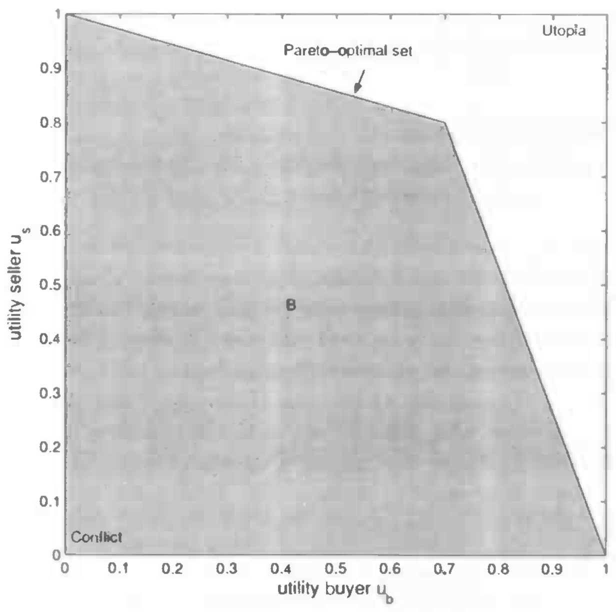


Figure 2.1: Pareto-optimal set, feasible set, utopia and conflict point

2.1.2 Pareto Optimality

Now we have a measure of individual performance, defined by the utility, we can define efficiency more concretely. To do this, we use the utility space representation as given in figure 2.1. In this figure, the utility of a buyer (b) is plotted against the utility of the seller (s), using a linear utility function $U_i : \Omega \rightarrow [0, 1]$, where Ω is the set of possible contracts or *allocations* and $i \in I = \{b, s\}$. In this figure, *utopia* is the situation where both agents achieve their highest possible utility. The *conflict point* is the utility the agents get when there is no deal made: Throughout this thesis we define this to be the origin[3]. The set of *possible outcomes*, or feasible allocations, is every possible combination of utilities, the whole graph in this case.

Of course, not every outcome of the possible outcome space can be achieved. To define the constraints on this set, we will first define what is rational for an agent:

Definition 2.3 (Individual rationality) *An agreement is said to be* individually rational *if it assigns each agent a utility that is at least as large as the agent can guarantee for itself from the conflict point.*

Applying this to figure 2.1 ensures only contracts in the plotted quadrant will be individually rational for both agents.

On the other hand, the set of possible outcomes is constrained by the *Pareto efficient set*. This is a widely used efficiency measure which defines the points in which an improvement of utility for one agent necessarily means a decrease in utility for the other agent. To get a formal definition, we define contracts to

[3]Note that, since the utility function is ordinal, this does not imply a loss of genericness.

consist of the issue-set $J = (1, \dots, m)$, and the set of agents to be $I = (1, \dots, n)$. An allocation for issue $j \in J$ is denoted as x_j. *Pareto optimality* is then defined as:

Definition 2.4 (Pareto optimal) *An outcome* $(x_1, \dots, x_m)$ *is* Pareto optimal *if there is no other feasible allocation* $(x'_1, \dots, x'_m)$ *such that* $u_i(x'_j) \geq u_i(x_j)$ *for all agents* $i \in I$ *and all contract-issues* $j \in J$ *and* $u_i(x'_j) > u_i(x_j)$ *for some* i, j.

An example of a Pareto optimal line is plotted in the graph, assuming both agents have linear utility functions and different weights over the issues. This line can be calculated by the weighted method (Raiffa, 1982). However, this requires the preferences of the agents to be public.

2.2 Negotiation Protocols

When looking in more detail to the related work on automated negotiation, we will start by discussing the work on negotiation protocols. Since we defined negotiation in this work as being bilateral encounters, in this section we will not focus on large scale society protocols, such as voting, auctions and market mechanisms. For a comprehensive review of these mechanisms, see (Sandholm, 1999).

The bilateral protocols we will discuss in this section involve two parties (for example a service supplier and a service consumer) and, generally speaking, an alternating offers protocol. This protocol is inspired by (Rubinstein, 1982), where the following bargaining situation is presented: two players have to divide a pie of size 1. Each has to make, in turn, a proposal how to divide it. After one party has submitted its offer, the other party has to decide whether to accept it, or to reject it and continue bargaining. This process is repeated until the parties come to a mutually acceptable agreement over the terms and conditions of a trade or one of the parties withdraws (typically because its negotiation deadline has passed). The protocol we use will be presented in more detail in section 3.1.

2.2.1 Cooperative Versus Non-Cooperative Models

A important distinction between the several models proposed is whether the agents are *cooperative* or *non-cooperative*. A cooperative situation is one in which the player can make binding agreements, as opposed to a non-cooperative game, in which they cannot. To make this clear, I will use the *Prisoner's dilemma*, one of the most common examples used in game theory, first analysed in 1953 at the Rand Corporation by Melvin Dresher and Al Tucker.

The story is as follows: There are two suspects of a crime. However, the evidence is not very convincing. Therefore, the prisoners are interrogated separately. If both confess, they each get 8 years in prison. If both hold out, however, they can only get punished for a lighter crime, for which the sentence is 1 year in prison. Finally, if one of the two confesses, while the other holds out, the confessor gets released, while the other gets the maximum punishment of ten years. For the normal form of this game see table 2.1.

The difference between cooperative and non-cooperative games is the difference between whether the two prisoners can talk to each other before making

		B	
		Confess	*Hold out*
A	*Confess*	(-8,-8)	(-10,0)
	Hold out	(0,-10)	(-1,-1)

Table 2.1: The prisoner's dilemma: Shown are the payoffs for (A, B)

their decision. The environment is called cooperative when the agents can make binding agreements. In non-cooperative environment, such agreements will not be possible, or will not be binding (in which cases the two prisoners might for example agree to confess, but when the time comes don't confess after all).

These two approaches to game theory are both interesting in their own right. Cooperative game theory frequently appeals to efficiency (the pareto-efficiency measure discussed earlier is an example of this) and fairness. Also, a lot of research has been done on bargaining games, e.g. by Nash (1950) and Shapley (1953). Non-cooperative games take a more fundamental stance, focusing on equilibrium strategies given a complete description of the rules.

2.2.2 Cooperative Models

Cooperative models are also known as axiomatic models. Axioms in this case refer to desirable properties of the solutions, where a solution consists of a set of strategies which are in equilibrium. This solution focuses on the desired properties of the outcome alternatives instead of process of the negotiation. Thus, the strategy used is not of great importance, it is the preference structure of the game that determines which contracts are feasible (Binmore, 1992). In this section, one of the most influential protocols derived from a cooperative bargaining situation is presented. After that, two models derived from the cooperative methodological stance are derived. First, the pioneering work of Rosenschein and Zlotkin (1994) is discussed, which applied cooperative game theoretic models and mechanism design to the design of different negotiation protocols for different domains. Second, a classic example of a concrete implementation is the *contract net protocol*, which can be used for the distribution phase of cooperative problem solving.

Alternating Offers Protocol

An influential body of work in the development of negotiation protocols, emerged from the field of cooperative game theory, is the work of Rubinstein (1982). He describes the *bargaining problem* as follows:

> Two individuals have before them several possible contractual agreements. Both have interests in reaching agreement but their interests are not entirely identical. What "will be" the agreed contract, assuming that both parties behave rationally? (Rubinstein, 1982)

In this work, a strategic approach is adopted. Two players have to reach an agreement on the partition of a good. Each player has to make a proposal in turn as to how it should be divided. Proposals can be either accepted or rejected, after which the negotiation continues. Further it is assumed that time is valuable (e.g. there is a bargaining cost per time step of the form $y - ct$) and that a player needs a compensation for a delay of each time step (e.g. there is a discounting factor of the form $y \cdot \delta^t$). The analysis which follows assumes complete information: the players have a correct and full knowledge of the preferences of the opponent. Since this is not applicable to the domain we are interested in, we will not review this analysis. However, the protocol itself is used in much of the work on automated negotiation, and will be used in adapted form in this thesis as well (see section 3.1 for details).

Domain Theory of Negotiation

The interest of the *domain theory of negotiation* lies in the engineering of protocols which are self-enforcing. In other words, agent designers should use these protocols since deviating from such a protocol would be irrational. The properties of such protocols, like efficiency, stability and symmetry, are then analysed, leading to three different domains of the negotiation problem:

i) *Task oriented domain:* The activity of the agents can be defined in terms of a set of tasks that it has to achieve. In principle, the task can be performed by the agent itself. However, cooperation may relieve the task.

ii) *State oriented domain:* Each agent wants to move the world from an initial state to a goal state. Due to limited resources, this may give rise to conflicts, which have to be resolved through negotiation.

iii) *Worth oriented domain:* The agents assign a worth to each potential state, which captures its desirability for the agent. This function allows agents to compromise on their goal, sometimes increasing the overall efficiency of the agreement.

For all these domains protocols are designed which are self-enforcing.

Within the scope of this thesis, all agents will have a utility function which maps their preferences to a value. Since this preference relation is complete (as per definition 2.1), the utility function also will be complete. Thus, each possible state (e.g. contract) has its own worth (e.g. its utility). Hence the work in this thesis falls into the worth oriented domain.

The model of Rosenschein and Zlotkin applied cooperative game theoretic models towards the design of models and mechanism design to computational agents. It made a first step in developing a general theory of negotiation. However, the model has a few drawbacks. First, it makes the assumption that agents have symmetric abilities, meaning that all agents are capable of performing all the necessary tasks at the same costs. In most domains however, this is not very realistic. Generally, costs do differ for certain actions between agents, especially when other variables, like demand, are not necessarily the same. Furthermore, most negotiation exist because an agent can not perform the service under negotiation itself, whereas the negotiation partner can. Thus, the agents will generally not have the same abilities.

Second drawback of this method is that the negotiation is assumed to be over one issue (worth in the case of the worth domain). As argued before, in most real cases, the object under negotiation will consist of multiple issues, and the protocol described will therefore not be applicable in most cases.

Finally, the agents have to be fully cooperative in showing their incentives before the negotiation starts. In this phase, the optimal strategy is implemented by the protocol designer. This can be difficult to achieve, because the agents are afraid of being exploited, even in a pre-negotiation phase, as argued before.

Contract Net Protocol

In a slightly different setting as this work, the Contract Net Protocol (CNP) has been developed for decentralised task allocation (Smith, 1988). Its significance lies in that it was the first work that uses negotiation processes and thereby uses mutual cooperation to solve a problem.

The method described a network consisting of nodes: the contract net. These nodes dynamically appoint a manager and a contractor, where the former is responsible for monitoring the execution of processing the results of a task and the latter is responsible for performing the task itself. Whenever a task has to be allocated, the node which generated this task announces it, and becomes the manager of this task. All other nodes evaluate their interest in this task, and propose a bid for the performance of it. The manager selects the node with the most interesting bid and assigns the task to this node (e.g. assigns the contractor). By this means, the procedure of the negotiation is made explicit. Furthermore, commitments are made explicit by the agreement between the manager and the contractor. Since the final manager-contractor commitment is a mutual selection process, cooperation after the deal is assured.

It must be noted that the original work of Smith consists only of the theoretical system architecture. However, (Sandholm, 1993) extends this with a formalisation of the bidding and awarding decision process. In this work, agents locally calculate their marginal costs for performing a set of tasks. The choice of a contractor is then solely based on these costs. Furthermore, some extensions on practical limitations are solved, like the clustering of several tasks.

Although the protocol has been used successfully in several domains, like job dispatching among computers and scheduling of appointments between agents, the protocol has a number of limitations when regarding the domain described in this work. First of all, full cooperation between the agents is needed when setting up the network. Second, the negotiation phase consists of just a single offer of all parties. In non-cooperative settings, negotiation often consists of an iterative process of offers and counteroffers to reach more efficient outcomes.

2.2.3 Non-Cooperative Models

Instead of looking at the solution, non-cooperative models is a field within game theory which is more interested in the processes during the game . The aim is to find a suitable pair of equilibrium strategies. It is often used to model bargaining situations (see e.g. (Binmore, 1992)), in which the process of the negotiation is analysed. An important issue in this work is the amount of information available about the opponent. When this information is complete, the seminal work of Nash (1950) provides a solution for the problem. However, the agents are assumed to be highly rational.

When looking at incomplete information settings, much work has been done in the field of optimal strategies. Theoretical solutions have been provided for incomplete information about deadlines (Sandholm and Vulkan, 1999), reservation prices (Fudenberg, 1985) and discounting factors (Rubinstein, 1985). However, all this work assumes that the beliefs of an agent regarding the strategy the other player is following are common knowledge, and agents are in principle computationally unbounded. It is this set of assumptions which makes this work hard to use in practise.

Non-Cooperative Task Allocation

A more practical solution is given by Kraus (1997), which recommends the application of game-theoretic techniques as a basis for the agents' interaction protocols. More specifically, an extension of Rubinstein's alternating offers protocol (see section 2.2.2) is presented, in which the world state and formal definitions are modified to provide a solution to task- and resource-allocation. Also, the need for searching appropriate strategies by the automated negotiators as opposed to defining these within the protocol explicitly, is acknowledged, specifically pointing out the need for low complexity since the agents are bounded in their computational power.

Furthermore, the utility-function, which is in the game-theory literature always assumed to be provided, is made concrete. These utility functions should comply with the following assumptions:

i) Disagreement is the worst outcome

ii) The resource is valuable

iii) Time is valuable

iv) Stationary: the preference relation between two goods is independent of time

v) Increasing loss: the loss associated with the time of agreement is an increasing function with regard to the utility of the contract (i.e. when the utility of a given contract s is greater than the utility of r, the loss over time associated with these contracts will be greater for s than for r)

These assumptions imply that the discount associated with time or resource will be embedded within the utility function. Finally, the alternating offers protocol is extended with the option for the agents to opt-out.

Although this work offers a very useful extension to the theoretical background of non-cooperative game theory, it does not meet the requirements of this work. The utility functions will be very domain-specific and necessarily complex due to the explicit inclusion of all the assumptions given above. Therefore, to design such a utility function is not an easy task for someone who is not an expert in utility theory. Instead, in this work we are looking for a more generic utility function, independent of time and resources. This dependence of time or resources can later be built in by separate discount functions.

2.2.4 Persuasion Protocols

Another class of negotiation protocols extends the communication with argumentation. In this form of negotiation, agents generate and exchange arguments to back up or justify their negotiation stance. By this means, information about the opponent is acquired by the feedback this opponent gives on previous proposals. This feedback can take the form of counter-proposals, in which the agent offers an alternative contract, or a critique, which comments on parts of the proposal the agent likes or dislikes (Parsons et al., 1998). Generally speaking, the initial situation in a persuasion dialogue is a conflict of points of view. To this end, persuasion deals with a set of questions at issue (propositions) toward which the parties have different points of view. This could for example be whether an issue should be included in the contract or not. The main goal for the agent is a resolution of the conflict by persuading the opponent's to take over its own point of view (Walton and Krabbe, 1995).

As can be seen, this technique will be used mainly in cases where agents have to achieve a goal which they cannot or prefer not to achieve on their own. These goals may for example be in conflict, in which case the agents have to bargain about which agent may achieve which goals. Successful persuasion of the opponent could prove useful within this scenario to convince the opponent of your point of view. Another example is a scenario in which the agents need the help of each other to achieve certain goals (e.g. Parsons et al. (1998)). Persuading each other to a certain plan could in this case improve efficiency. However, it is also applied to design systems for automated electronic purchase. For example, as a result of persuasion the preferences of an agent might change (think of a smooth-talking car-salesman which stresses convincingly the importance of an air-conditioning). Within this field, recent work had sought to define precisely the protocols specific to such argumentation-based interaction (e.g. Amgoud et al. (2000)). By using such protocols, participating agents can assert arguments in the dialogue.

However, the definition of such a protocol only presents an agent with the framework for an argumentation based negotiations. To supply a concrete implementation of agents using this protocol, McBurney et al. (2002) couple such a protocol to a model of consumer purchase decision making taken from marketing theory. This model is coupled with a more simple model for the seller. An entirely syntactical interaction protocol is then defined, ensuring the protocol and its rules are verifiable on the basis of the actions of the participating agents.

The protocol fulfills several of the requirements of this work. It assumes for example that the services under negotiation consist of multiple issues. Furthermore, the process of the negotiation is central in this work. Also, the fact that resources are bounded is not made impossible. However, there are a few major drawbacks to this method. The first problem is implied by the nature of persuasion. Persuasion operates primary over goals. In many domains however, for example the commercial domain, the goals do not have to be similar at the end of the negotiations; a buyer conceding on the price does not change its goal to minimise expenditure. Thus, although it might be of secondary importance to negotiation in some cases, it is not the goal of negotiation per se, and in some cases even irrelevant. As Walton and Krabbe (1995, p. 72) put it: "It would be a mistake to think, however, that persuading the other party that your point of view is right (true, based on logical reasoning or good evidence) is the primary goal of negotiation."

A second problem with the method is that, because of the inverifiability of semantic elements, no participants to a dialogue can know with certainty what another participants really believes. Therefore, McBurney et al. (2002)

assume that the agents are completely honest in their statements. However, in a competitive domain this is not tenable.

2.3 Negotiation Objects

The second broad topic for research are the *negotiation contracts* which specify the range of issues over which agreement must be reached. Within this area, the number of issues over which agreement must be reached is an important topic (see section 2.3.1). The work we will review here is the influential work of Fatima et al. (e.g. Fatima et al. (2002, 2003a,b)), which provides a formal analysis of the optimal outcomes of single-issue as well as multi-issue negotiations (section 2.3.2).

2.3.1 Number of Issues

In the simplest case, agents are negotiating over one issue, for example the price of a house, and will have strictly opposing interests. Think in this case of a negotiation about the buying/selling of a commodity, where the buyer wants the price to be as low as possible, whereas the seller wants to sell it for the highest price possible. Given these extreme values for all agent, known as the *reservation prices*, the zone of agreement (or bargaining surplus), is defined by the zone between these values; e.g., the zone between the maximum price of the buyer and the minimum price the seller is willing to accept. Now, the goal of the negotiations is to divide this bargaining "surplus". Hence the negotiations are *distributive* or competitive: A gain for one party always creates a loss for the other party. Such situations are called *zero-sum* games within game theory.

Negotiation becomes especially interesting when a conflict must be resolved over multiple issues. In this case, when the one party gets more this does not necessarily mean the other party gets less. Hence, the parties are not strictly competitive, but they can enlarge the pie that they eventually will have to divide (Raiffa, 1982). Since several issues, possibly with different importance attached to these issues by the players, are involved, trade-offs become an option. In this case, a player concedes on one issue, while demanding more on another issue. While the utility of the player remains the same, the utility of the opponent might increase. This point is illustrated in more detail in section 3.3.1. The possibility for such a mutual gain, thus in fact enlarging the pie that has to be divided instead of dividing it as in distributive bargaining, is why this sort of negotiation is called *integrative*, or even *side-by-side problem solving* (Fisher et al., 1991).

2.3.2 Optimal Outcomes

Most influential work within this area analyses the optimal outcomes of single-issue negotiations and multi-issue negotiations (in which the issues are negotiated sequentially).

Single-Issue

First, a competitive negotiation between a buyer and a seller over the price of the good or service is considered. Each agent has time constraints in the form of a deadline and a discounting factor. Fatima et al. (2003b) examine a range of negotiation scenarios in which the amount of information that agents have about their opponent's parameters is systematically varied, both symmetric, in which case both agents are uncertain about the same parameters, and asymmetric, where the uncertainty on a certain parameter is one-sided. For each scenario the equilibrium solution is determined and its properties studied. This shows that for each possible scenario, a unique solution is given by the optimal strategy.

However, Fatima et al. (2003b) consider only single-issue negotiations. Thus, since in this work it is assumed that the service under negotiation has multiple issues, it can not be directly applied here.

Multi-Issue

In other work, multi-issue negotiations are considered in which the issues are negotiated one by one. A key question that arises in this case is the order in which the issues are negotiated or the *agenda*. When the agenda is *endogenously* defined, i.e. the agents are allowed to decide which issue they will negotiate next during the process of negotiation, a unique equilibrium exists (Fatima et al., 2002). Whereas Fatima does not define how this agenda is agreed upon, Walton and Krabbe (1995) note that persuasion might be involved in setting up such an agenda for negotiation (Walton and Krabbe, 1995).

In Fatima et al. (2003a) the agenda is defined partly endogenously and partly exogenously (i.e. before the negotiations). However, this scenario requires a mediator to identify the optimal scenario, and is therefore not compatible with our requirements.

Given this, we consider agendas that are set entirely exogenously and in which all the issues are considered simultaneously. The optimal solutions in the work mentioned above can thus not be used.

2.4 Reasoning Models

Finally, the most important topic for this research is the *reasoning model* which provides the decision making methods the agents employ to compute their negotiation moves. Within this area, the importance of learning from past negotiation experiences was first recognised in Sycara's work on the PERSUADER system (Sycara, 1989) which modeled an iterative process of multi-issue negotiation (see section 2.4.1).

After this, a variety of learning techniques have been used to try and improve the effectiveness of the agents' negotiation capabilities. For example, genetic algorithms have been used to discover effective negotiation strategies (see section 2.4.2 for an overview). Probably the most widely used paradigm is *Bayesian learning*. The models derived from this technique are described in section 2.4.3.

Probably the work that is most closely related to ours is that of Soo and Hung (2002). In this work, a reinforcement learning algorithm is presented that learns to propose a solution that is more likely to be accepted by the opponent than the previous offer. Specifically, offers that are rejected by its opponent are treated as negative instances for the learning algorithm, while counter-proposals from its opponents give a positive reward. However, their model does not combine the learning model with a reasoning model that models the properties of the problem domain as the fuzzy similarity method of Faratin does (see section 3.3 for details). This means their model is less robust when the prediction about the preference weights is not completely precise.

2.4.1 Persuader

The PERSUADER system was developed to resolve adversarial conflicts in the domain of labour relations (Sycara, 1989), which can be multi-agent, multiple-issue and single or repeated encounters. It uses past agreements between similarly situated parties to suggest proposals that might succeed in the current negotiations. A mediator engages in parallel negotiations with the parties, either to change the proposal to something that is acceptable, or to attempt to change the belief of the disagreeing parties using persuasive argumentation. To this end, it uses *case-based reasoning* together with an analytical method called *preference analysis*.

In this context, negotiation is viewed as an iterative process of diminishing the distance between the goals of the agents. Furthermore, agents must have the capacity to receive feedback on the quality of their plans as well as predict and evaluate the quality of their own plans. In addition to this deliberative component, the agent should be reactive to cope with a changing environment. Finally, since agents are not willing to abandon their own goals, persuasion should be used to convince them to do so.

The system models both the process of negotiation as well as the multi-issue requirement. However, in our domain, mediation is undesirable because of the competitive nature of the encounter and the desire to keep information private. Furthermore, persuasion can not be used in all cases and is not a primary goal of negotiation, as argued in section 2.2.4, and will therefore not supply a generic solution to the problem presented in this work.

2.4.2 Genetic Algorithms

An often used paradigm for the analysis of negotiation strategies are genetic algorithms (e.g. Gerding and van Bragt (2003); Oliver (1997)). In this research, agents are modelled as chromosomes and the parameters of the negotiation model are genes in the chromosome. By evolving these agents, the benefits and drawbacks of a number of negotiation strategies are assessed.

For example, in the work of (Gerding and van Bragt, 2003), a system is described for bilateral negotiations in which the agents are generated by such an evolutionary algorithm. More specifically, the model of the agent consists of the absolute offers and thresholds (i.e. determines whether an offer of the opponent is accepted or not) for each round in the negotiation process. These agents start a bargaining process against all other agents. The fittest agents are subsequently selected to form a new generation by using mutations and crossover in their genotypes. The protocol used in the negotiations is the alternating offers protocol. Furthermore, the fairness of the payoffs is taken into account, and in another extension, a competitive market with multiple bargaining opportunities.

However, this method typically focuses on finding the optimal strategy for choosing the offers and counter-offers, whereas we want an explicit reasoning model about the opponent.

2.4.3 Bayesian Learning

There is a vast amount of research in using Bayesian learning within the reasoning model of a negotiation agent (e.g. Bui et al. (1995); Zeng and Sycara (1998)). In this line of work, the estimates of the probability of a set of hypotheses about the opponent's preferences or reservation prices are produced, given the previous negotiation encounters. As a simple example, consider the problem that there are two possible hypotheses about the preference function of a negotiation opponent, say A and B. The Bayesian method, rather than choosing between A and B, gives some weight to each based on their likelihood. This likelihood will depend on how much the known data supports each of the two preference functions.

More specifically, suppose that we have a set of samples, $D = \{d_j\}$, and a series of classes $C = \{c_i\}$. Given a sample d_j, the probability of that sample belonging to class c_i can directly be derived from Bayes' rule:

$$P(c_i|d_j) = \frac{P(d_j|c_i)P(c_i)}{P(d_j)} \tag{2.4}$$

Thus, the probability of class c_i (one of the hypotheses) given the observation equals the probability of the sample given that the hypothesis of being in class c_i is correct, times the probability of encountering class c_i, divided by the probability of encountering this sample. It can be seen from this equation that the agent has to have a priori knowledge about the probability distribution of the likely outcome of the negotiation ($P(c_i)$).

In more detail, the method of (Zeng and Sycara, 1998) tries to learn information about the opponent's reservation prices RP_s. The buyer needs a partial belief about the possible RP_s, represented by a finite set of hypotheses $H = H_i$ (where for instance $H_1 = 100$, $H_2 = 90$, etc.). Furthermore, it needs a priori knowledge about the probability of these hypotheses (e.g. $P(H_1) = 0.2$, $P(H_2) = 0.34$, ...). Along with domain knowledge, new offers of the opponent can enable the player to acquire new information about RP_s in the form of posterior subjective evaluation over H_i (see equation 2.4). Thus, an offer combined with domain knowledge such as "in our business people will offer a price which is above their reservation price by 17%" will lead to an enforcement of the correct hypothesis.

This example shows some significant drawbacks of Bayesian learning. First, the a priori probability evaluation of the hypotheses is, due to the private nature of the information needed to compute this, difficult to provide. Also, the domain knowledge will be hard to acquire as precisely as indicated here. Often, a party will have "but an imprecise feel for their own reservation price" (Raiffa, 1982, p. 46), and, even more, the strategies and domain knowledge used are not at all clear to the parties themselves (Raiffa, 1982). Even if assigning such an a priori distribution was practically possible, the number of negotiations needed to get a more precise probability distribution might be too much in a domain in which encounters are not necessarily frequent. This is a problem emerging from the fact that individual opponents are modelled. However, in the KDE-method presented in this work, the information is completely acquired out of the data available. Therefore, such an a priori probability over hypotheses is not needed.Furthermore, as opposed to modelling a specific opponent, this work models the strategy (dependent of the agent's preferences) with respect to the provision of a particular service (see section 3.2 for more details). By this means, the method is more generic, and thus less computationally complex.

Chapter 3

Theoretical Models

In this chapter the theoretical models that underpin this work will be described. To this end, we distinguish three theoretical models. First, the negotiation model used will be described (see section 3.1). Most importantly, this model captures the negotiation protocol which will be used. Also, the formal definitions of the used parameters are given.

The second model described is the kernel density estimation paradigm (see section 3.2). This statistical method is used to learn and approximate the preferences of the opponent. Within this model lies the novelty of this research, since it has never been used for learning the preferences of a negotiation opponent. We will explain the method in section 3.2.

To evaluate the efficacy of the KDE-method, we couch it in the context of making trade-offs: Giving in on one issue, while simultaneously demanding more on another issue. This strategy is specifically suited for testing our method because it can profit extensively of knowledge about the opponent's preferences. More concretely, we use the algorithm described by Faratin (2000) which evaluates possible trade-offs based on fuzzy similarity. This algorithm is presented in section 3.3.

3.1 The Negotiation Model

Before evaluating the negotiation model, let us first start with defining some of the necessary parameters: If I is a pair of (self-interested) negotiating agents ($I = \{a, b\}$), let i ($i \in I$) represent a specific negotiating agent, and J ($J = \{1, \ldots, n\}$) be the issues under negotiation in a given encounter. For each issue j ($j \in J$), every agent has a lower and an upper reservation price, respectively min_j and max_j. These values represent respectively the value which is the best reasonable value expected and the worst value still acceptable for the agent. To illustrate this, suppose an agent wants to buy a house. His upper reservation price will be the maximum price it can accept; any price that is higher than this value represents a situation for the buyer that is worse than no agreement. The lower reservation price is the price he thinks is the lowest reasonable price for the house. This will often be his opening bid. Mutatis mutandis for the seller of the house.

These values result in a domain for each particular issue and for each agent i:

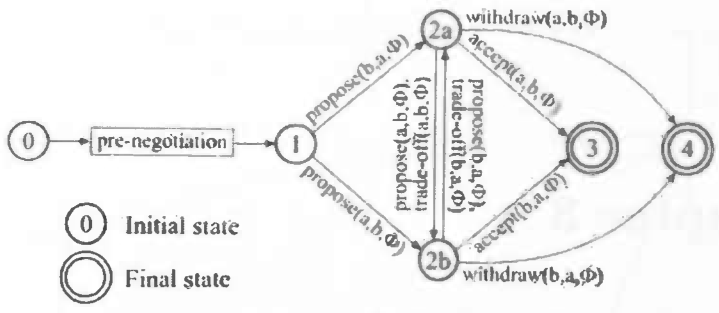


Figure 3.1: State diagram of the protocol used, adapted from (Faratin, 2000, fig. 4.1).

$\mathcal{D}^i_j = [min^i_j, max^i_j]$[1]. In all negotiation interchanges, every issue gets assigned a value x_j by agent i which is in the set $\mathcal{D}^i_j$. Note that the issues are held constant during the negotiation, and are assigned a fixed value (as opposed to contingent contracts where e.g. the final prices are determined to *ex-ante* agreed price-quality pairs).

To evaluate a value of an issue, each agent i has a scoring function over its domain: $V^i_j : \mathcal{D}^i_j \rightarrow [0,1]$ which assigns a valuation to every possible value x_j. Finally, each agent has a weight vector over the issues, representing the relative importance it attaches to the issues, where w^i_j is the importance agent i attaches to issue j. We assume these weights are normalized (i.e. $\forall i \in I : \sum_{1 \leq j \leq n} w^i_j = 1$). Thus, the utility of agent i over a *contract* x, a set of values for all issues, can be defined as:

$$u^i(x) = \sum_{1 \leq j \leq n} w^i_j V^i_j(x_j) \tag{3.1}$$

Now, if all this information is known to both agents, the *Pareto-optimal* set can be calculated by both of them (Raiffa, 1982). However, in e-commerce settings, several of the assumptions needed for this calculation are not tenable. First, an agent will not give out information about its reservation prices, the weights over the issues and its utility function because doing so would enable it to be exploited. Second, each agent will have distinctive reservation prices (whereas in Raiffa (1982) these were defined to be equal for all agents). The zone of possible agreements $\mathcal{Z}$ we will use is thus defined as the intersection of the individual domains: $\mathcal{Z}_j = \bigcap_{i \in I} [min^i_j, max^i_j]$. Finally, the resources, and especially the time available for the negotiations, will be bounded. Specifically, we assume each agent has a (hard) deadline, denoted as t^i_{max}, by when it must have completed the negotiation.

The negotiation protocol is a two-agent variant of Rubinstein's model of *alternating offers* (Rubinstein (1982), see also section 2.2.2). Specifically, let $x^t_{a \rightarrow b}$ be the offer of agent a to b, at time t and $x^t_{a \rightarrow b}[j]$ denote the value of issue j of this offer. Note that in this model time is discrete. The protocol used is depicted in figure 3.1. The negotiation is set up by the pre-negotiation protocol, which establishes a connection between the agents and defines the issues under

[1]Note that when these intervals for the negotiation participants do not overlap, the zone of agreement (as described in 2.3.1) will be empty and thus an agreement will not be possible.

negotiation. The agent who has first turn is chosen randomly. After the first offer, at every timestep the agent who received the last offer decides whether to accept the offer, propose a counter-offer or trade-off, or withdraw from the negotiation. This continues until the reaction is one of the communication particles $\{accept, withdraw\}$.

The agents decide which of the alternatives to choose from the following definition (taken from Faratin (2000)), where $x^{t'}_{a\to b}$ is the calculated counter-offer:

Definition 3.1 *Given an agent a and its associated scoring function V^a, the* ***interpretation*** *by agent a at time t' of an offer $x^t_{b\to a}$ sent at time $t < t'$, is defined as:*

$$I^a\left(t', x^{t'}_{a\to b}\right) = \begin{cases} withdraw & If\ t' > t^a_{\max} \\ accept & If\ u^a(x^t_{b\to a}) \geq u^a(x^{t'}_{a\to b}) \\ x^{t'}_{a\to b} & otherwise \end{cases}$$

The scoring function over the issues, as required by equation 3.1, is given by the distance to the worst bid acceptable to this agent, relative to its range of acceptable values. Hence this scales the acceptable bids for the agent to the domain $[0, 1]$:

$$V^a_i(x_j) = \begin{cases} \frac{x^t_{a\to b}[j] - \min^a_j}{\max^a_j - \min^a_j}, & \text{if increasing} \\ \frac{\max^a_j - x^t_{a\to b}[j]}{\max^a_j - \min^a_j}, & \text{if decreasing} \end{cases} \tag{3.2}$$

where increasing and decreasing refer to the direction of change in score with increasing value of the issue.

To calculate the counter-offers used in definition 3.1, a range of different strategies could be deployed. Given the time-constrained nature of our domain, however, we use the family of *polynomial strategies* (as advocated by Faratin et al. (2002)), which models the fact that agent will change their behaviour as the deadline approaches. More specifically, the offer on issue j generated by agent a at time t will be calculated with the following expression:

$$x^t_{a\to b}[j] = \begin{cases} \min^a_j + (1 - \alpha^a_j(t))(\max^a_j - \min^a_j) & \text{if increasing} \\ \min^a_j + \alpha^a_j(t)(\max^a_j - \min^a_j) & \text{if decreasing} \end{cases} \tag{3.3}$$

where the function α is a polynomial function dependent on the time remaining, and β is the strategy parameter, defining the form of the function:

$$\alpha^a_j(t) = k^a_j + (1 - k^a_j)\left(\frac{\min(t, t^a_{max})}{t^a_{max}}\right)^{\frac{1}{\beta}} \tag{3.4}$$

It can be seen from this function that β determines the degree of curvature of the function (as can be seen in figure 3.2). The parameter k^a_j determines the value of issue j to be offered in the first proposal by agent a. For simplicity, within this we assume this parameter to be 0, resulting in the final equation:

$$\alpha^a_j(t) = \left(\frac{\min(t, t^a_{max})}{t^a_{max}}\right)^{\frac{1}{\beta}} \tag{3.5}$$

This family represents an infinite number of strategies. To make a better classification possible, they are generally divided into three sets:

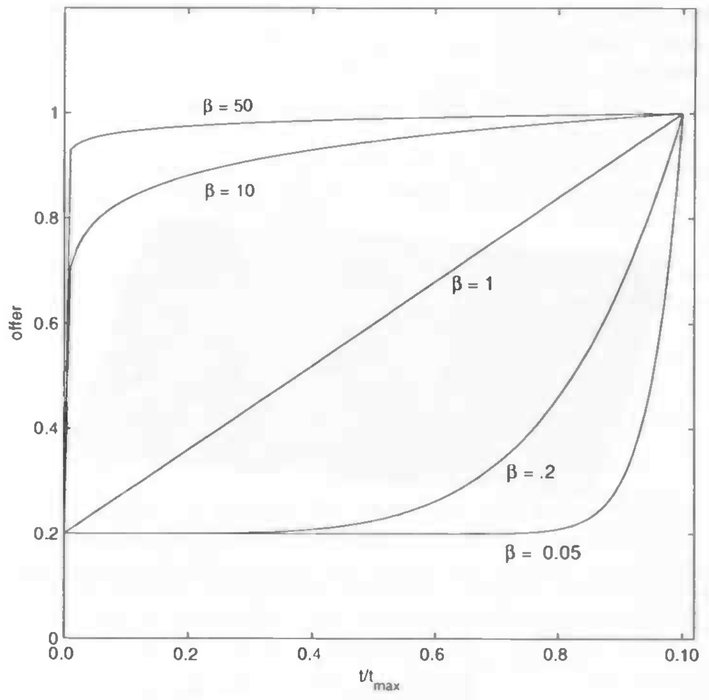


Figure 3.2: Polynomial strategies and their subfamilies illustrated. X-axis depicts the time relative to the deadline. Upper two lines are *conceder*, lower two lines *boulware* strategy. The line in the middle ($\beta = 1$) is a linear strategy

Boulware[2]strategy When $\beta < 1$ the tactic generates offers close to the lower reservation price. When the deadline comes near, it concedes quickly towards the upper reservation price (see the lower two lines in figure 3.2.

Conceder strategy When $\beta > 1$ the tactic quickly concedes towards reservation price, and stays firm to this during the rest of the negotiations (see the upper two lines in figure 3.2)

Linear When $\beta = 1$, the tactic concedes an equal amount with every offer.

3.2 Kernel Density Estimation

The only information we can definitely assume to be available to the agent is its negotiation history. This covers the offers and counter-offers of all its previous negotiations for a particular service[3]. Therefore, our aim is to obtain an estimate of the opponent's weights by only looking at this history. In particular,

[2]Lemuel Boulware was a vice-president of the General Electric company who rarely made concessions in wage negotiations; his opening-offer was what he deemed to be a fair opening offer and held firm.

[3]This history can be on a per agent basis or can cover all agents with which the modeller has interacted for the particular service in question.

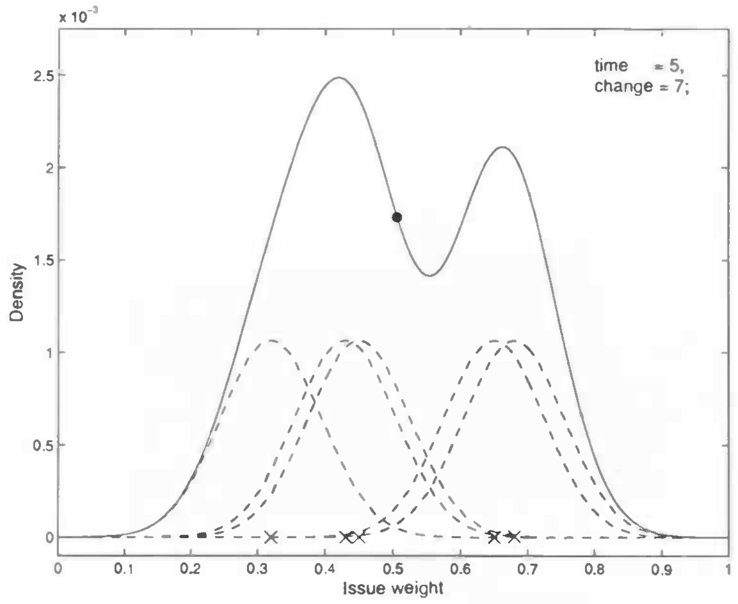


Figure 3.3: Kernel density estimation example. The dashed lines are the kernels and the solid line the estimate. The individual observations are marked with a cross and the predicted weight for the negotiation issue in question with a dot.

we consider the difference between the opponent's last two offers in a given encounter and try to find a relation between this difference and the weight the opponent places on various issues. For example, a relatively small change in the bid on one issue at the beginning of the negotiation process might indicate that it is more important to the opponent than the other issues. To give an concrete example, think again of the agent that wants to buy a flight ticket. When the price is the most important issue of this ticket for this agent relative to issues like for example class and time of flight, he will first try to concede on these less important issues before giving in on the price to get a good deal out of it for itself. Likewise, a relatively large concession towards the end of the negotiation might indicate that this issue is important and the opponent is performing a final concession on that issue to save the agreement. The buyer might for example have conceded less on the price of the flight ticket at this stage of negotiation, and will try to get the ticket by conceding relatively much towards its reservation price.

The method we will use to find this relation is *kernel density estimation* (for reasons outlined in section 1.4). In more detail, the basis of this method is the *kernel*: a function K satisfying $\int K(X)dx = 1$. Intuitively, these kernels can be seen as representing a "probability mass" of size $1/n$ (where n is the number of observations) associated with each data point, about its neighbourhood (Wand and Jones, 1995). This is illustrated in figure 3.3, where a two-dimensional estimate is based on five observations[4]. This example could be viewed as the

[4]It should be pointed out that we use just five observations here purely for clarity in illustrating how the kernel method works. Practical density estimation usually involves a much higher number of observations.

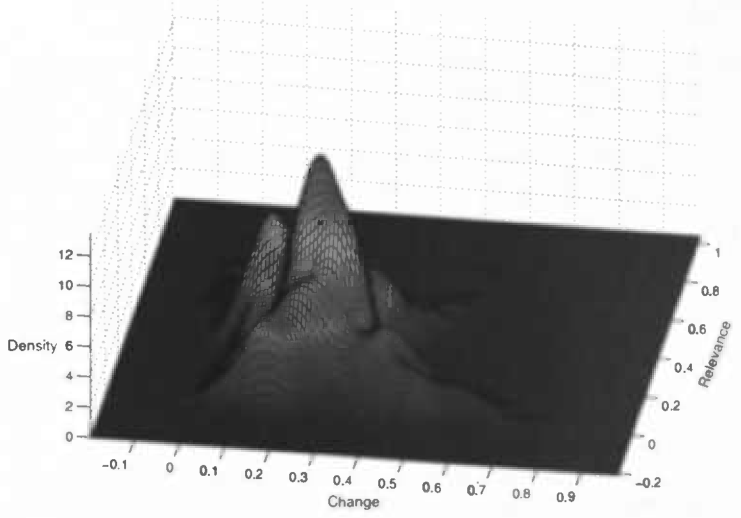


Figure 3.4: Kernel density estimation example: Density (z-axis) given a change between last two offers (on x-axis) and the weight of the issue (y-axis).

estimate of the opponent's weight for a particular negotiation issue, given a specific time in the negotiation process and the relative difference of the two last bids on an issue. In this case, as indicated in the top-right corner of the graph, we see that these observations took place at the beginning of a negotiation encounter (time = 5) and there was a relatively small change between two consecutive offers (7 percent of the total change). Here, the dashed lines are the *kernels*. It can be shown that the particular unimodal distribution used as a kernel does not degrade performance (Wand and Jones, 1995) and so we choose the $N(0,1)$ distribution for reasons of computational efficiency. The kernels are formed by centering a kernel at each observation (e.g. the difference between two consecutive offers and the believed weight of the issue). Note the kernels are scaled by the total number of observations and thus the value of the kernel estimate at point x is simply the sum of the scaled kernels. For example, in figure 3.3 it can be seen that there are no observations for a very low weight and therefore the density of 0.1 has a low value. On the contrary, there are relatively many observations around a weight of 0.4 and hence the probability estimate has a high value here. From this distribution the weight predicted for the negotiation issue in question is the expected value, using the estimate as a probability density function (indicated with a circle in the graph).

In this illustrative example we used two-dimensional kernels. However, the kernels we use in this research are in fact three-dimensional (the difference of two consecutive offers, the weight of the issue and the probability density of this weight given this difference). Furthermore, since we assume the opponent uses a time-dependent strategy, we expect the agents to behave differently over time. Therefore, for each time t, we make such a density estimate to predict the density of a certain weight given the relative difference between the last two offers. This leads to an estimate as in figure 3.4 per time-step, where the

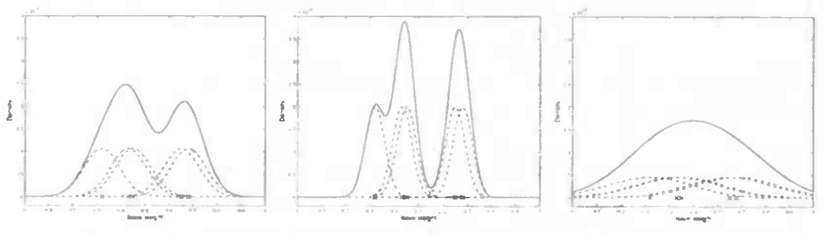

(a) $h = 0.075$ (b) $h = 0.040$ (c) $h = 0.175$

Figure 3.5: The influence of bandwidth on the kernel density estimate. (a) Shows the bimodal property of the data. In (b) the lower bandwidth results in higher and smaller kernels, leading to an overfitting of the data. (c) has a higher bandwidth, resulting in an underfitting; the clear bimodality in the data is smoothed away.

x-axis denotes the change between two consecutive offers, the y-axis denotes the weight of the issue and the z-axis the density at this point.

One of the most important issues in this work is the *bandwidth* or the amount of spread of the kernels, denoted by h. Statistically, this is the confidence in the observations, or the standard deviation of the gaussians. The effect of this parameter is illustrated in figure 3.5. In these graphs, the same observations are shown as in figure 3.3, and kernel density estimates are formed with kernels of varying bandwidth. Now suppose the estimate in graph 3.5(a) is indeed the best estimate. When the bandwidth of the kernels is taken too low, as in graph 3.5(b), the kernels become smaller and higher, since the confidence that the observations are correct is higher. This leads to an overfitting of the data, as shown by the higher modes and lower troughs and the division of the first mode into two separate modes. When the bandwidth of the kernels is taken too low, so a low confidence is placed in the correctness of the data, the estimate becomes oversmoothed, as seen in graph 3.5(c). This obscures the two modes completely.

To obtain the most optimal bandwidth, we use the *solve-the-equation* (STE) rule as suggested by Wand and Jones (1995) to calculate this bandwidth. This method takes an initial guess of the bandwidth and calculates a new value, using the error in the resulting density estimate, until the process convergences. The reasons for choosing this method are its good performance on our domain (due to the direct feedback from the results of previous values), and the ease of computation.

Since the negotiating agents have bounded computational resources, it is important to determine the computational complexity of KDE. To this end, when the probability density estimate is learnt, the prediction of a weight is a table lookup of constant time. The learning algorithm calculates this estimate by using a fourier transformation of the kernels. The complexity is determined by the use of a convolution, which can be performed in $\mathcal{O}(n \log n)$ time (where n is the sampling rate, taken to be 128) when using a fast-fourier transformation to calculate it (Kamen and Heck, 2000). This is also the complexity of the bandwidth estimation, since the error of the estimate is used in the computation and thus the estimate has to be formed, and the number of iterations proved to be of constant complexity when tested in practice.

3.3 Trade-Offs Based on Similarity Criteria

The importance of a mutually acceptable agreement is generally acknowledge within multi-issue negotiation, for example by (Walton and Krabbe, 1995, p.72):

> This aim [to maximise the share of some goods or services which are short in supply] is carried out by a process of self-interested bargaining where the strategy is directed to finding a compromise that will be attractive to both parties

To find such a compromise, trade-offs are considered as an import strategy (e.g. Raiffa (1982), Fisher et al. (1991)). However, since the preferences of the opponent are private information in most domains, finding such a trade-off is not trivial.

To solve this problem of finding a trade-off when agents have incomplete information about their opponent we exploit Faratin's algorithm (Faratin et al., 2002). This algorithm works by performing an iterated hill-climbing search in the landscape of possible contracts. The search starts at the opponent's last offered contract (where the opponent is denoted as b) and proceeds by successively generating contracts whose utility is progressively closer to the desired threshold of the agent making the trade-off. Note, since the agent wants to make a trade-off, he must have done a proposal before, of which the utility serves as this threshold. During this search the contract that maximizes the similarity to the last offer of b is used as the starting point of the next iteration. We will first present the rationale for using trade-offs. After that, the algorithm is presented in more detail.

3.3.1 The Rationale for Making Trade-Offs

Consider a situation in which two agents are negotiating over a service[5]. Agent a wants a high price and a long delivery time, whereas the preferences of agent b are strictly opposite. These preferences are shown in figure 3.6. The values of both issues of agent a are depicted on the lower x-axis and the left y-axis respectively. The values of agent b are depicted on the opposite axes. Within this box (called an *Edgeworth-box*) the iso-curves are drawn (solid lines for agent a, dashed lines for agent b): the sets of allocations with the same utility for this agent. Hence, agent a wants to move the outcome northeasterly, whereas agent b wants to move southwesterly. Thus, it can be seen that a is indifferent between contract P, R and S, whereas agent b is indifferent between R, Q and S.

Now suppose agent a proposes contract P. After b proposes Q, a can now, by conceding on issue 1 while demanding more on issue 2 or vice versa, respectively propose contract R or S. Note that these contracts have the same utility for it than P, as they all lie on the same (solid) iso-curve. However, since these contracts lie on an iso-curve of b that is more south-west than the original iso-curve, the utility for b increases, thereby making the contract more acceptable to this agent than P. In fact, as acceptable as Q, the contract offered by b itself.

[5]Example adapted from Raiffa (1982).

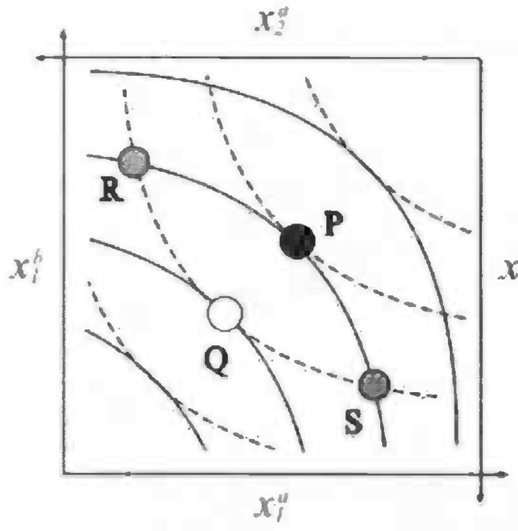


Figure 3.6: The concept of a trade-off illustrated. Depicted are the iso-curves for a (solid lines) and b (dashed lines), where a's preference is northeast, and b's preferences southwest

3.3.2 The Trade-Off Algorithm

How the algorithm works is illustrated in figure 3.7. Here, two agents are represented. The first agent, a offers contract x (which lays on iso-curve a). The opponent, b reacts with contract y. Agent a wants to propose a trade-off now. Hereto, it starts with the utility of contract y (according to its own utility function), and divides the difference with the utility of x in S steps. For each of these steps, N contracts on the required iso-curve are generated. Of these contracts, the one most similar to contract y is selected and used in the following step to generate new contracts.

More specifically, the algorithm thus consists of two steps, assuming that agent a has to make a bid:

i) Find the iso-curve for a, that is, the set of contracts $x^{t'}_{a\to b}$ which can possibly be proposed at time t', that have the same utility for agent a as its previous offer at time t (i.e. $x^{t}_{a\to b}$),

ii) Out of this set, take the contract that agent a *believes* is most preferable to the opponent, and send this as the trade-off counter-offer.

Both steps will be dealt with in turn in the following sections.

3.3.3 Contract Generation

The first step, to find a contract on a given iso-curve, is given as an algorithm in figure 3.8. In general, the algorithm generates a new contract by splitting the step gain in utility, E, randomly among the set of issues under negotiation. In more detail, the algorithm starts by initializing the domain of possible improvements per issue j, denoted as $\overline{E}_j$, in line 0. Hence since the maximal

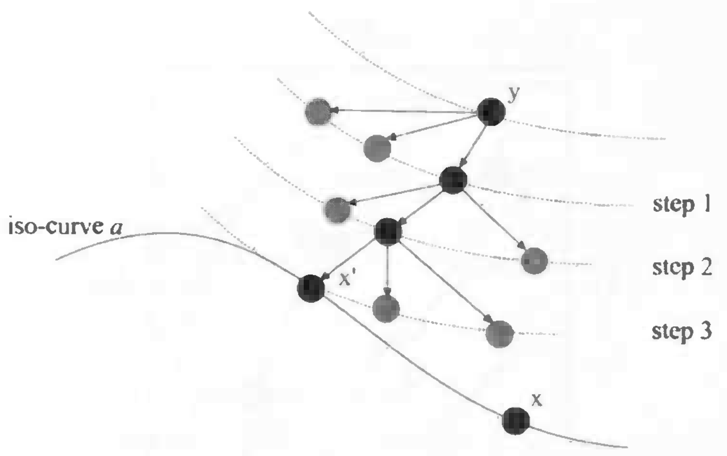


Figure 3.7: The trade-off algorithm

(unweighted) score on an issue is 1, this is the difference between 1 and the actual score of the contract that is being modified. The sum of these maximal gains weighted by the issue-weight determines the maximal utility-gain on the contract ($E_{\max}$).

Next, because the "consumption" of this utility gain has a random element (line 4), a small factor is included to guarantee convergence[6] (line 2). The `while`-loop that now begins, builds a matrix r in which per row the amount of "consumptions" is determined until the desired utility-gain has been reached. More specifically, this is done by letting each issue consume a random value out of the domain $\overline{E}_j$ calculated in line 0, further constrained by the remaining gain needed (line 4). The variable of the current total amount consumed (E_n) is then updated by the linearly weighted sum of the individual consumptions. Also, the domain of possible improvements per issue is accordingly updated.

When the total consumed utility is equal to E, the `while`-condition fails. The total consumption for each issue is then calculated by summing the columns (line 7). Finally, this new set of scores per issue is remapped to a contract by the inverse of the scoring function $V()$ (line 8). Note that when the desired amount of improvement (E) is larger than the maximal improvement possible ($E_{\max}$) an error is raised (line 9).

As presented here, it is assumed that the issues are quantitative in nature (e.g. price) with continuous domain values. This protocol can be easily extended to deal with qualitative issues as well (e.g. color) which have discrete domain values, as it is done by (Faratin, 2000). Since this does not change the algorithm structurally, in this work we only use quantitative issues.

It can be shown by a theoretical analysis that the average time the algorithm takes to complete is linear with respect to the number of issues in the negotiation (see (Faratin et al., 2000) for details of the proof). Given the aim of this research to respect the computational limitations of the agents, this is a highly desirable property.

[6]The convergence is asymptotic to the value $V(y) + E_{\max}$. Hence, if we had $E = E_{\max}$, reaching the iso-curve could not be guaranteed.

```
Inputs
    $y^t$    : Last best contract proposed at time $t$, consisting of the issue-values $[y_1^t, \dots, y_J^t]$.
    $E$      : Step utility increase.       $\{ E > 0 \}$
    $V()$    : Value scoring function.      $\{ V() : y^a \rightarrow [0,1] \}$
    $w$      : Weight vector of $j$.        $\{ \sum_{j \in J} w_j = 1 \}$

Outputs
    $y^{t+1}$ : Output contract.            $\{ V(y^{t+1}) = V(y^t) + E \}$

Begin
0.  For all $j \in J$ do
1.      $\overline{E}_j = [0, 1 - V_j^t(y_j)]$
2.  End for
3.  $E_{\max} = \sum_j w_j \max \overline{E}_j$
4.  $\delta = 0.01 \cdot E_{\max}$
5.  If $(E_{\max} > E + \delta)$ then
6.      $k := 0$; $E_n := 0$;
7.      While $(E_n < E)$ do
8.          $k := k + 1$; For all $j \in J : r_j^k = 0$ ;
9.          For all $j \in J$ do
10.             If $(E_n < E)$ do
11.                 $r_i^k = \min(\text{random}(\overline{E}_j), \frac{E - E_n}{w_j})$
12.                 $E_n = E_n + w_i \cdot r_j^k$
13.                 $\overline{E}_j = [0, \max(\overline{E}_j) - r_j^k]$
14.             End if
15.         End for
16.     End while
17.     For all $j \in J$ do
18.         $\overline{E}_j = \sum_{i=0}^{k} r_j^i$
19.         $y_j^{i+1} = V_j^{-1}(V_j(y_j^i) + E_j)$
20.     End for
21. Else
22.     Raise error: *No step can be performed*
23. End if
End
```

Figure 3.8: Contract generation part of the algorithm

3.3.4 Contract Selection

In the second step, the selection of the best generated contract on the iso-curve, *fuzzy similarity* is used. The algorithm thus tries to find the deal that is most "similar" to the previous offer. The rationale behind this is that a deal which is similar to an acceptable offer of your opponent has a reasonable probability of being acceptable itself.

In this context, the notion of similarity between two valuations of issue j, $x_j, y_j \in \mathcal{D}_j$ uses a criteria evaluation function $h : \mathcal{D} \rightarrow [0,1]$ which maps the value of the issue to a valuation between zero and one[7]. When comparing two values for a specific issue, the issue-similarity is defined by comparing the values of the function h:

$$\text{Sim}_j(x_j, y_j) = 1 - |h(x_j) - h(y_j)| \tag{3.6}$$

The similarity of two contracts is then defined by the sum of the issue-

[7]The function used here has the form of a sigmoid:
$h(x) = \frac{1}{\pi} \tan^{-1} \left[\left(\frac{2|x - \min|}{x - \min} \cdot \left| \frac{x - \min}{\max - \min} \right|^{\alpha} - 1 \right) \tan \left(\pi (\frac{1}{2} - \varepsilon) \right) \right] + \frac{\pi}{2}$

similarities weighted by the opponent's weight of that issue:

$$\text{Sim} = \sum_{j \in J} w_j^b \cdot \text{Sim}_j(x_j, y_j) \tag{3.7}$$

This results in the following formal definition of the algorithm:

Definition 3.2 *Given an offer x from agent a to b, and a subsequent counter offer y from agent b to a, and given that $\theta = u^a(x)$, agent a defines its trade-off proposal with respect to y as:*

i) $\text{iso}_a(\theta) = \{x \mid V^a(x) = \theta\}$.

ii) trade-off$_a(x, y) = \arg\max_{z \in \text{iso}_a(\theta)}(Sim(z, y))$.

To increase the exploration of the space of possible deals, the algorithm starts at the utility of the contract of the opponent (y), and takes S steps in increasing the value of θ, until the utility of the previous proposal x is reached. The number of elements generated in step i is defined by N.

3.3.5 Overview

The trade-off algorithm of Faratin is chosen because it has a number of desirable properties: (i) its complexity is linearly proportional to the number of issues under consideration and (ii) by using the notion of fuzzy similarity the uncertainty of an agent's belief over the preferences of the other agents are modelled as fuzzy relationships between values of the domain (and not the other agent's actual preferences). Hence, the algorithm models the domain of the issues under consideration (the problem domain), instead of the individual agents (recall the discussion at the end of chapter 2).

While this trade-off algorithm has been shown to be effective in a number of scenarios (Faratin, 2000), however, a major shortcoming is that the weights the opponent used in calculating the similarity between two offers (see equation 3.7) are private information and thus unknown to agent a. Given this, the aim of this thesis is to see if the KDE-method outlined in section 3.2 is effective at learning this information using only the negotiation history.

Thus, by extending a solid basis with a separate learning model, as per section 3.2, we obtain a hybrid algorithm which makes the trade-off algorithm more adaptive and robust.

Chapter 4

Experiments

In order to employ a formal and systematic evaluation of the work in this thesis, a set of experiments has been designed to evaluate and measure the performance of the proposed method. In particular, a variety of statistical inference methods can be used to verify hypotheses from the samples of data gathered during the experiments (including regression, analyis of variance (ANOVA) and path analysis). For purposes of this work, we will extensively use ANOVA, which tests the variance between groups. It is chosen because it is a general and robust technique: "analysis of variance is very robust against violations of the normality and equal variance assumptions, especially if the group sizes are equal." (Cohen, 1995).

We will start by laying out the methodology to be used in evaluating the experiments in section 4.1. After that, we will evaluate the experiments. The general aim of these experiments is to examine the effects of using a weight vector as predicted by kernel density estimation, on the performance of the trade-off algorithm. Since the utility of this trade-off remains the same for the agent itself (see definition 3.2), we measure performance in terms of the opponent's utility of the proposed trade-off contract. Specifically, we start by looking at a single offer to investigate how the prediction of the weights by KDE influences the performance of the trade-off algorithm (section 4.2). After that, we investigate the effects of the KDE-method on a complete negotiation process by analysing agents which use the predictions in their negotiation strategy (section 4.2.4).

4.1 Evaluation Methodology

4.1.1 Experimental Methodology

To get the desired systematic evaluation of the experiments, the experimental framework proposed by Cohen (1995) will be used. In concrete terms, this means that the variables of interest are identified and observed during or after the application of a particular process. These observations are then analysed statistically against hypotheses formed a priori, resulting in accepting, rejecting or revising these hypotheses. The experiments in this thesis can thus be classified as exploratory studies: "*[studies that] yield causal hypotheses that are tested in observations or manipulation experiments*" (Cohen, 1995, p. 7). In this class,

Dependent Variable	Notation	Value
Number of steps	S	40
Number of children per step	N	100
Location of optimal discrimination	α	1
Spread of optimal discrimination	ε	0.1
Number of simulation runs over which the data is collected		50
Weight vector of buyer	w_b	$\{0.5, 0.1, 0.05, 0.35\}$
Weight vector of seller	w_s	$\{0.1, 0.5, 0.25, 0.15\}$

Table 4.1: Dependent variables

designing an experiment is categorised by the following items, which are dealt with in turn (Excelente-Toledo, 2003):

i) Experimental procedure,

ii) Dependent or environmental variables,

iii) Independent or experimental variables,

iv) Hypothesis formation.

Experimental procedure. This describes the experimental procedure in detail. By making this explicit, spurious effects can be determined more easily. In the context of this thesis, the procedure captures the agent's knowledge (e.g. whether he knows the parameters of the opponent fully or partly) and the particular process (e.g. whether the experiments consider one single offer or a complete negotiation process). In particular, the processes which model the performance of the trade-off mechanism extended with the kernel density estimation method are of interest in this thesis.

Dependent variables. These describe the attributes and properties of the environment in which the experiment is performed. These variables remain static during the experiment. In our experiments, these variables consist in the first place of the parameters used by the trade-off mechanism. As described in section 3.2 these are the number of steps to take from the opponent's contract to the trade-off contract to propose (S), and the number of children to calculate each step (N). We will use the values of Faratin et al. (2002): 40 steps using 100 children per step. More steps or more children are shown not too improve the performance of the algorithm.

The similarity function $h(x)$ used in equation 3.6 is parametrized to be a linear function. In particular, the α, which determines where the maximal discrimination power lies, is set to 1, resulting in optimal discrimination power in the middle of the domain. The amount of discriminability within the domain of the reservation values (ε), is set to 0.1, resulting in an optimal discrimination within the reservation values, but almost no discrimination outside this domain. The remaining variables consist of the variables needed by the environment. Since there is a random factor in the trade-off algorithm, the proposals are calculated 50 times (a significant sample size considering 0.95 confidence level, with confidence interval of 15%). Finally, the weight used by respectively the buyer and seller are chosen in such a way that each agent has different preferences over the issues. Table 4.1 summarizes these variables and their assigned values. Unless stated otherwise these values are used in all the hypotheses tested in these experiments.

Independent variables. These describe the data of interest; that is, the data that should be examined and measured from the experimental environment. In the context of this thesis, this is the performance of the agent (defined in different ways for the various experimental processes). We will use the mean performance over all simulation runs.

Hypothesis formation. This involves identifying a claim to direct the experimentation. Specifically, a hypothesis is formulated based upon the experimental variables to test the execution of the experimental procedure given the particular dependent variables (Excelente-Toledo, 2003).

4.1.2 Evaluating Hypotheses

After laying out an experiment as described above, the hypothesis must be tested on its correctness. To this end, the experiment is performed and the values of the experimental variables are collected. The general idea then consists of inferring the probability that the hypothesis is correct. Formally, the procedure is the following (Excelente-Toledo, 2003):

i) Formulate a hypothesis (the null hypothesis, represented by H_0),

ii) Show that the probability of obtaining a given result given H_0 is above a certain threshold. This threshold is known as the significance level (the confidence level is defined as $100 \times (1 -$ significance level$)$),

iii) Conclude H_0 under this level of confidence

In our experiments we will generally test the performance of various agents in varying environments (as measured by the mean performance over all runs). In this case, step 2 in the procedure described above consists of statistically comparing these mean performances against each other. *Analysis of variance* is used for this due to the method being general and robust. Specifically, it tests the hypothesis that the means of several groups are equal, given the variation within the groups. This is done by analysing all possible interactions among them.

The method rests on some assumptions. First, the population distribution from which the groups are drawn are assumed to be normal. Second, these distributions are assumed to have the same variance. Third, the error components of the data in the groups are assumed to be independent. The method, however, is very robust to violations of the first two assumptions, especially if the group sizes are equal (Cohen, 1995). Although the third assumption is more stringent, it is not a problem since the experimental design does not imply a dependency between observations in the group, thereby also making the errors components independent[1].

However, since this only tests the hypothetical question (i.e. whether there is a difference), further analysis is necessary to determine the specific relationship

[1]For a more formal revision of the analysis of variance method, see e.g. Cohen (1995).

among the groups (i.e. what are the exact differences between the groups, which group is better than the others). For example, to test whether the performance of groups A, B, C and D is similar, H_0 is rejected if the ANOVA shows there is no difference. However, this does not say anything about the performance of A relative to B, A relative to C, and so on. The procedure for testing this consists of running a post-test on a case by case basis. It forms clusters of groups which have statistically homogeneous means with an associated value that indicates the degree of confidence with which the group was built. The specific testing method used in this thesis is Tukey's *honestly significance difference* (HSD) method. This method was chosen because it lies in the middle of the spectrum of alternatives; between the least significance difference method, which yields more spurious differences, and the Scheffé tests, which can make it difficult to demonstrate differences between means (Cohen, 1995; Excelente-Toledo, 2003). To obtain the results from an ANOVA, we use the implementation of *Matlab*[2] which performs the anova itself and compares the scenarios with a post-hoc test.

4.2 Single Offer Experiments

The aim of these experiments is to examine the effects of using a weight vector, as predicted by kernel density estimation, on the performance of the trade-off algorithm. Since the utility of this trade-off remains the same for the agent itself (see definition 3.2), we measure performance in terms of the opponent's utility of the proposed trade-off contract. Specifically, in this section we start by looking at a single offer to investigate how the prediction of the weights by KDE influences the performance of the trade-off algorithm. The next section investigates the effects of the KDE-method on a complete negotiation process by analyzing agents which use the predictions in their negotiation strategy.

In these experiments a specific instance of a negotiation session is considered, resulting in a single offer – counter-offer pair. Due to the number of private and thus unknown dependent variables, we do not expect the prediction of the KDE to be completely precise. Therefore, we analyse the results when a random perturbation is added to the real weight of the opponent (section 4.2.1). After that, we explore to what extent a priori knowledge of the opponent is necessary for good performance of the trade-off algorithm. We do this by looking at the concrete effects of using KDE's formed under different levels of knowledge about the opponent (section 4.2.2). Finally, to test whether using KDE increases the performance with respect to the method proposed by Faratin, we compare the performance to a situation in which uniform weights are used (section 4.2.3). All experiments described here follow the settings as described in 4.1 unless stated otherwise.

4.2.1 Random Perturbation of Real Weights

For this experiment a negotiation between two agents (that generate their offers according to equation 3.3) was undertaken to get a range of plausible contracts (using the model described in section 3.1). Specifically, the contracts consist of four issues of unequal weight that remain fixed throughout the negotiations (with the weights as described in 4.1). For simplicity it is assumed for all issues

[2]Matlab v.6.5, release 13, The MathWorks Inc.

that an increase of value results in a utility gain for one agent ("the seller", or agent s), and a utility decrease for the other agent ("the buyer", or agent b).

From this negotiation, we take two consecutive proposals: a bid x of b and a counterproposal y of s. Using these two proposals, a trade-off for the buyer is calculated (as per definition 3.2). Specifically, we consider two environments (A, in the beginning of the negotiations, and B, later in the negotiations) in which the utility of b varies. In environment A, the utility of agent b is high (because it has not conceded much with its time-dependent strategy), hence there will be fewer contracts on its iso-curve. In environment B, however, the utility of b is much lower (because time has passed and it has conceded utility). This results in more contracts on the iso-curve, and, consequently, more opportunity to improve the utility of agent s.

The experimental variable in this experiment is the correctness of the weights used for the opponent (as used in equation 3.7). In the first scenario, the agent has perfect knowledge (scenario *full*). The second scenario involves a small perturbation of the opponent's real weight vector to evaluate the effect of a small error in the prediction of the weights. This perturbation is chosen uniformly out of the domain [-0.05, 0.05] (scenario δ = *0.05*) and [-0.1, 0.1] (scenario δ = *0.1*). For reference purposes, the results of two benchmark situations are added. First, to check what happens if the prediction is completely arbitrary, a randomized weight vector (satisfying $\sum_J = 1$) is used in the scenario *ran*. In the second situation, we do not use any prediction about the opponent at all, but just choose a child randomly in each step of the trade-off algorithm to recurse into (scenario *no sim*). This gives us a means of comparing our results to the case in which similarity criteria are not used at all.

Since the change of the bid for the opponent depends on multiple unknown variables (as can be seen in equation 3.3), we cannot expect the prediction to be completely precise. The objective of this experiment is therefore to analyse the effects of using such inaccurate information. It is important to note that here, in contrast to Faratin et al. (2002), we do not make the assumption about the preservation of ordinality in the opponent's weights. Thus, we will evaluate the following hypotheses:

HYPOTHESIS 1. *A small error in the prediction of the opponent's weights will not significantly degrade the performance when compared with the perfect information case.*

HYPOTHESIS 2. *The prediction of the weight vector does not have to be ordinally perfect to improve performance compared with the case of using no knowledge about the opponent.*

The results of this experiment are presented in figure 4.1, where the utilities of the agents are plotted against each other. The line joining (0,1) to (1,0) is the Pareto-optimal line, calculated using the weighted method (Raiffa, 1982) (see 2.1). In this case an ideal trade-off would be on the Pareto-optimal line. Therefore, the closer our trade-offs are to this, the better they are. Since environments A and B are reached using non-optimal methods, however, we will just look at the order of the contracts, and not the concrete utility reached.

In more detail, the figure illustrates a part of a negotiation process. To evaluate the performance of the KDE-method, the following setting was used. Consider two agents, a service consumer (the buyer) and a service supplier

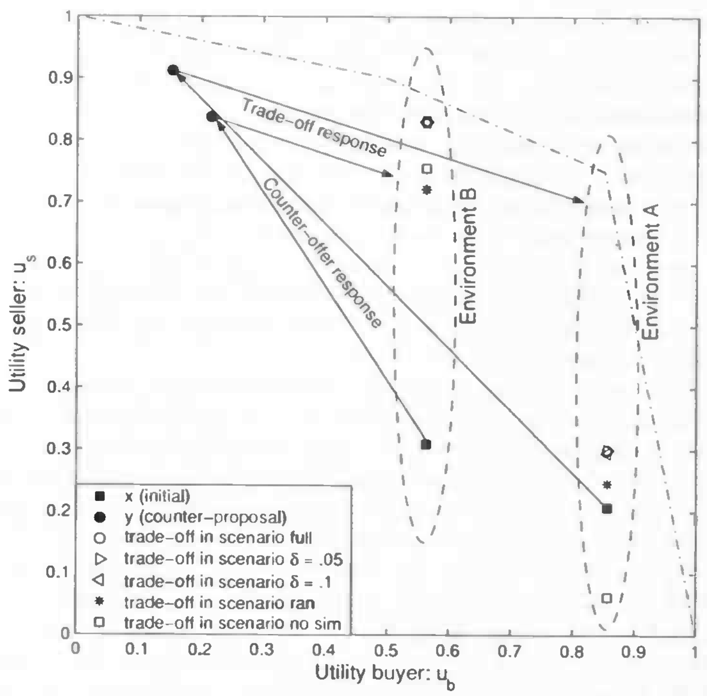


Figure 4.1: The mean utility of the proposed trade-offs using perturbed weights.

(the seller), who want to reach an agreement on the terms of a contract by negotiation. After a few proposals and counter-proposals, the buyer again makes a proposal and receives a counter-proposal from the seller. It now decides to make a trade-off. It is the performance of this trade-off that is evaluated.

In the figure, this process is depicted as follows: the filled squares are the contracts proposed by the buyer in preparation of the trade-off, plotted for both environments A and B. As indicated by the arrows, the counter-offers of the seller are plotted as filled circles. After receiving this counter-offer, the buyer calculates a trade-off response (for the different scenarios mentioned above) which are plotted within the different environment ellipses.

As can be seen, for both environments, the performance of using weights that are perturbed does not decrease the performance of the trade-off algorithm. This is shown by an *analysis of variance* (*ANOVA*) extended with a *HSD* test as post-hoc test, which indeed shows that in both environments there is no significant difference between the proposed contracts[3] when using full knowledge of the weights of the opponent or a perturbed weight. In more detail, the ANOVA shows that the probability of the 5 classes as coming all from the same class is zero for both environments. When looking at the post-hoc test, it can be seen that in environment A there are three classes. The worst performing class is formed by the *no sim* scenario. The scenario *random* performs significantly better. The highest mean utility, however, is a third class, in which the scenarios *full*, $\delta = 0.05$ and $\delta = 0.1$ perform all the same with a mean utility of 0.3. In

[3]Using a confidence level of 0.05.

environment B, the same pattern is observed for the latter three scenarios. However, the performance of scenario *ran* and *no sim* do not differ significantly now.

This outcome can be explained by the combination of the KDE-method with the similarity-based method. Specifically, the resulting algorithm uses both the prediction of the weights by the KDE-method and the fuzzy similarity as a basis for computing trade-offs. This combination leads to a robust overall performance because errors in the prediction are compensated for by the fuzzy similarity method, and the fuzzy similarity method is enhanced by better information about the opponent.

The contracts produced by the control methods are significantly worse than the performance of the full knowledge (as expected). If we look at the *no sim* case, the contract to use in the next iteration is selected randomly from the children generated, which leads to poor performance because no attempt is made to maximize the opponent's utility. In the *ran* case, the weights of the opponent are randomized which means that although the algorithm does perform a maximization, it maximizes the situation in which the weights are different from their actual values.

Note that given the weight vectors used, the ordinality of the weights can change in both environments due to the perturbation. However, in the $\delta = 0.05$ scenario, only the order of the first and fourth issue can change, whereas in scenario $\delta = 0.1$ also the third issue can be at different places in the ordering. In either case, however, the absence of a significant difference between these scenarios indicates that this does not make a significant difference.

Overall, these results indicate that both hypotheses can be accepted. The algorithm is robust enough to achieve similar performance to the full information setting when the information is less than perfect. Moreover, this result is independent of the preservation of ordinality of the weights. When taken together, these results show that the prediction by the KDE does not have to be completely accurate to improve the performance of the trade-off algorithm.

4.2.2 Kernel Density Estimates of Weights

After having shown that approximate values for negotiation preferences can be used to produce effective trade-offs, the next step is to verify that the KDE-method can indeed give a prediction that is sufficiently accurate to use in the trade-off algorithm. Thus, instead of the real weights with a random perturbation, we now use the prediction of the opponent's weights under different levels of knowledge (which are represented by different kernel density estimates). The key difference between these estimates is the information about the opponent that is known to the agent. Since the opponent's bidding strategy is influenced by the deadline of the negotiation, $t^a_{\max}$, the reservation prices, RP, and the strategy parameter β (as can be seen in equation 3.3), these are the parameters varied in the different scenarios presented below. By this means we analyse the effect of the amount of knowledge, and therefore the precision of the prediction, on the efficiency of the trade-off algorithm.

In most domains, for example in e-commerce, the strategy parameter will be most difficult to obtain information about. It is hard to deduce from past experiences and may change for each individual encounter. Therefore, we start in the scenarios *strat20* and *strat100* with the strategy as a private parameter

	β	$t^s_{\max}$	RP
strat20	$[[0.1, 0.3]^{0.01}, [4, 6]^{0.1}]$	65	fixed
strat100	$[[0.1, 1]^{0.01}, [1, 10]^{0.1}]$	65	fixed
bdl	$[[0.1, 1]^{0.01}, [1, 10]^{0.1}]$	[30,100]	fixed
all	$[[0.1, 1]^{0.01}, [1, 10]^{0.1}]$	[30,100]	normal

Table 4.2: Variable initialisation when learning the kernel density estimates.

and the reservation price and the deadline as public information. In this case, a KDE is used which has learnt from opponents with varying βs, chosen uniformly out of the domain. The difference between the two is that the size of this domain for *strat100* is much bigger (two times one hundred possible values versus two times twenty possible values with the same stepsize). This means the buyer is much less sure about the opponent's strategy in *strat100* than in *strat20*.

In the next scenario *β-and-deadline*, or *bdl*, again the strategy parameter is varied, but now the deadlines are uncertain as well. This reflects the fact that agents will have deadlines which change over time, depending on the starting time of the negotiation. This is particurly the case in an e-commerce setting, but shall be applicable to most domains. Therefore, a probability distribution based on past experiences may not be particularly useful and it will be hard to get. Finally, in *all*, all the parameters are uncertain. It should be noted however, that in all cases the reservation prices are known to be in a normal distribution (scaled to a lower $\overline{\text{rp}}$ of 10, an upper $\overline{\text{rp}}$ of 20, and a standard deviation of 1.5)[4]. See table 4.2 for the specific values of all the scenarios. Note that the domains of the uniform distributions are made discrete and the stepsize, if it is not equal to one, is stated as a superscript.

Given this, the particular hypotheses we sought to evaluate here are as follows:

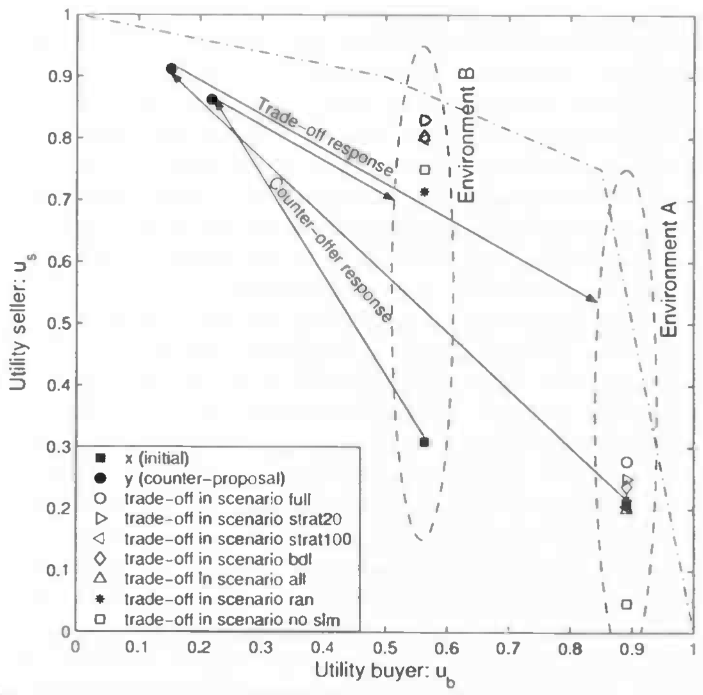
HYPOTHESIS 3. *Using kernel density estimates will result in a higher utility than not using any knowledge about the opponent.*

HYPOTHESIS 4. *The more knowledge about the opponent that is used in the kernel density estimation, the higher the achieved utility will be.*

The first hypothesis captures the expectation that when using KDE to predict the weights of the opponent, the performance of the proposed trade-offs will improve. If this were true, it would justify the use of KDE as a learning method that can be added to the trade-off algorithm. The second hypothesis reflects the expectation that KDE should perform better when there is more information about the opponent. In particular, we expect the performance of the trade-off algorithm to improve when the predictions are more precise, and thus we expect a higher utility when the agent has more a priori knowledge about its opponent.

The results are presented in figure 4.2, in the same manner as described in section 4.2.1. Thus, the buyer proposes a contract (depicted with a filled square) and gets a counterproposal of the seller (the filled circle indicated with the arrow marked 'counter-offer response'. The buyer now calculates a trade-off

[4]This standard deviation is based on the price-distribution of products often sold on the internet – like digital cameras and laptops – on price comparison sites (e.g. *kelkoo*: www.kelkoo.co.uk) and auction-sites (e.g. *eBay*: www.ebay.co.uk).



	Without KDE				*With KDE*			
	x	*full*	*rand*	*no sim*	*strat20*	*strat100*	*bdl*	*all*
A: u_s	*0.206*	0.278	0.217	0.047	0.249	0.205	0.237	0.201
B: u_s	*0.309*	0.830	0.714	0.750	0.830	0.799	0.804	0.802

Figure 4.2: The mean utility of the proposed trade-off using different knowledge levels.

using the settings of the various scenarios, resulting in trade-off options within the dashed ellipses. Since some of the mean utilities of these trade-off options lie close to each other, the results are also presented in the accompanying table. As can be seen, especially in environment A, all values are close to each other. This can be explained by the fact that there are fewer contracts on the iso-curve compared to environment B. Using an *ANOVA* shows that the probability of these results to show when there would be one class is zero. The post-hoc test shows that with a mean utility of 0.05, the contracts reached when not using similarity criteria performs far worse than the rest. The next class is formed the by random scenario, *strat100* and *all*. In fact, the random strategy performs surprisingly well in this environment; the results of the ANOVA show that it overlaps with both this class as by the next, in which the other KDE-strategies (*bdl* and *strat20*) lie. Having full knowledge of the parameters of the opponent is the best performing scenario, significantly improving the utility compared to the other scenarios. These results can be explained by the fact that the space of improvements is small due to the comparatively small number of points on the iso-curve, and, in turn, the error when using incorrect information is smaller.

In the second environment, the differences are more visible. Again, the performance using the correct weights of the opponents performs best. This time, however, the prediction of the KDE using *strat20* is good enough to result in the absence of a significant difference compared to the complete knowledge scenario. The *no sim* situation is, as before, worse than all the outcomes using a KDE, although it outperforms the *random* situation now. We believe the two most likely scenarios in e-commerce are *bdl* and *all* (where the knowledge about the opponent is minimal) and, as our results show, in these cases KDE performs well, indeed nearly as well as the situation where full information is assumed. This indicates that the predictions made by the KDE in such situations are sufficiently precise to use in the trade-off algorithm.

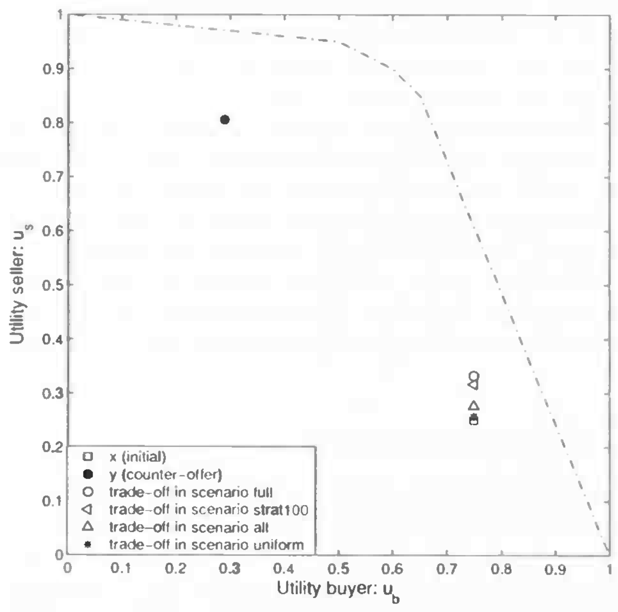
In general, our results can be explained by the precision of the prediction of the weights. When this information is completely correct (as in situation *full*) the performance is best. When the knowledge about the opponent degrades, the prediction gets less accurate, and the performance of the algorithm gets worse. We show that the kernel density method performs at least as well as having no knowledge at all when there are few contracts on the iso-curve. However, the results are far better when more possible trade-offs are available. This indicates that hypothesis 3 can be accepted. Moreover, the amount of information used while learning the KDE does not necessarily improve the performance (as is the case in environment A). However using partial information does affect the performance (as shown in environment B). This implies hypothesis 4 has to be accepted, although the similarity basis of the trade-off algorithm ensures the effect is not as strong as expected (because it makes the trade-off algorithm more robust to errors in the prediction).

4.2.3 Uniform Weights

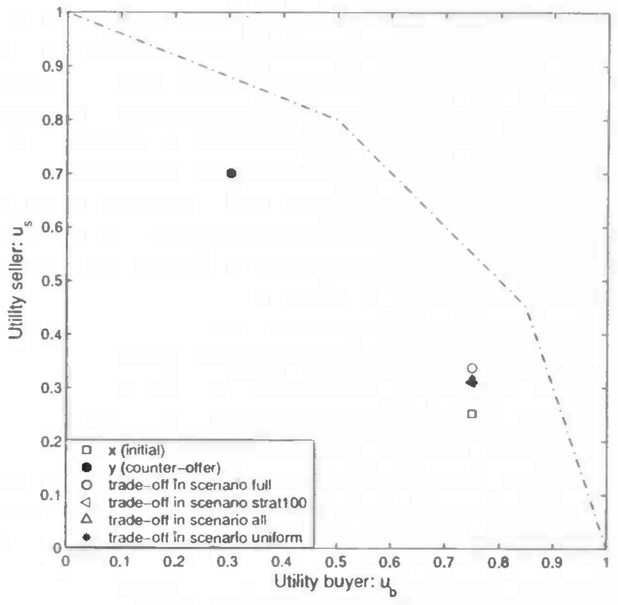
Now we have shown that KDE can be used effectively in combination with the trade-off algorithm, we want to show that our method outperforms the main method advocated by Faratin. Following his experiments on the performance of the similarity-algorithm, he concluded that the best policy for computing trade-offs is to assign uniform (equal) weights to all decision variables which can be updated by some learning rule (which is unspecified) (Faratin et al., 2002). To this end, we want to see how this works compared to using a KDE. Note that when the results are analysed, it must be taken into account that when the real weight vector of the opponent is close to uniform, the results of using a uniform weight vector are obviously going to be better than when the weight vector of the opponent is far from uniform. To account for this, we consider two environments; one with a uniform opponent and one with a strongly skewed weight vector (i.e. the former uses $w = \{0.2, 0.3, 0.15, 0.35\}$ as its weight vector, whereas the latter uses the weight vector $w = \{0.85, 0.05, 0.05, 0.05\}$).

To make a better comparison between these two environments, we want the utility of the starting contract to be similar. However, since the weight vectors used for the seller differ, this results in different contracts.[5] To achieve this we set the weight vector for the buyer as before. Since we showed in the last experiment that the influence of the level of knowledge is limited, the results will only be compared between the situations *strat100* and *all*, and to the performance when having all knowledge (*full*). Specifically, we hypothesize that:

[5]Note that a different weight vector also results in a different Pareto-optimal set.



(a) The performance of the scenario in which opponents' weights are assumed to be uniform when the opponent has non-uniform weights: the result is depicted with an asterisk.



(b) The performance of the scenario in which opponents' weights are assumed to be uniform when the opponent has indeed (almost) uniform weights: again, the result is depicted with an asterisk.

Figure 4.3: The mean utility of the proposed trade-off compared to the performance using uniform weights.

HYPOTHESIS 5. *Using kernel density estimation to predict the weights of the opponent will result in a utility at least as good as using uniform weights.*

The results of the experiment are shown in figure 4.3. In the upper graph, the opponent uses the skewed weight vector. In this case, an analysis of variances shows that the four different groups are significantly different from one another. In particular, assigning uniform weights performs significantly worse than using the kernel density estimate for the group *all*. This is as expected due to the discrepancy of the real weight vector and the uniform vector used in the computations. When the weight vector of the opponent is near uniformity, as plotted in figure 4.3(b), assigning uniform weights performs as well as using the kernel density. This is as expected because the real weights and the weight vector used do not differ from each other by much. Thus hypothesis 5 is accepted.

4.2.4 Complete Negotiations

Having shown that using KDE in combination with Faratin's trade-off algorithm improves the performance in the single-offer case, we will look at the results of using KDEs in a complete negotiation encounter. Henceforth, we will use the KDE-method in combination with a *meta-strategy* (i.e. a strategy of choosing which offer-generation strategy to use) which depends extensively on the trade-off algorithm. Our aim is to determine whether the more efficient individual trade-off offers will also result in a better negotiation outcome over a complete encounter.

The particular meta-strategy we will use is the *smart*-strategy (Faratin et al., 2002), which is shown to have a good performance against a variety of other strategies. The smart-strategy consists of deploying a trade-off mechanism (as per section 3.3) until the agent observes a deadlock in the closeness of two offers. Such a deadlock situation is observed when the similarity between these two offers is smaller than δ (set to 0.05 in this experiment). If this occurs, the algorithm starts looking for a contract with a value of the previous offered contract $V_b(x)$ reduced by a predetermined amount η. The average closeness is measured by the similarity between the last two bids of the agent. This leads to the following bidding strategy for agent a:

```
if $(|\text{sim}_b(x_{t-1}) - \text{sim}_b(x_t)| < \delta)$
    choose $(x_{t+1} : x_{t+1} \in X \wedge u_b(x_{t+1}) = u_b(x_t) - \eta)$,
else
    choose $(x_{t+1} : x_{t+1} \in X \wedge u_b(x_{t+1}) = u_b(x_t))$.
```

where η is the decrease in utility when a deadlock occurs (also set to 0.05 in this experiment). Note that this method always uses the trade-off strategy, and therefore makes extensive use of the prediction of the opponent's weights by the KDE-method.

In the experiment, both agents use the above trade-off strategy, the seller has full knowledge about the opponent, and the amount of knowledge the buyer has about the seller is varied (in a similar way to section 4.2).

The performance of the strategy is tested in two ways. First, the utility of the deal is calculated by taking the product of the utilities of the agents, where a higher product means a better performance. By looking at the product, we make sure that the deal is symmetric: there is not one agent performing better at the

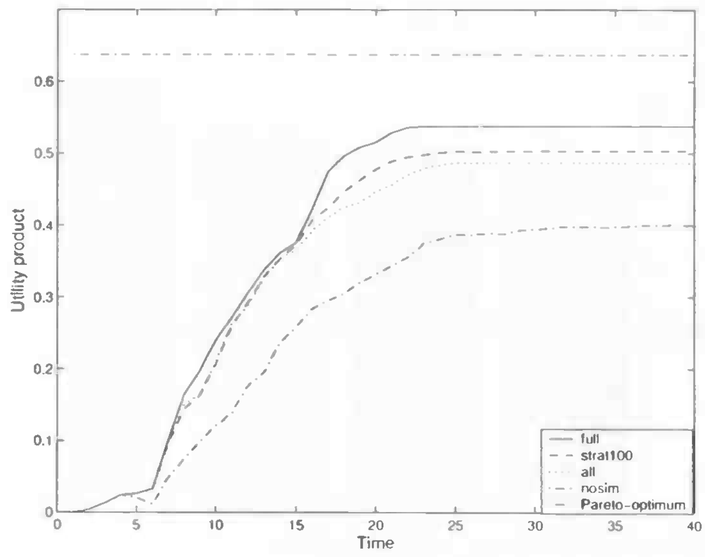


Figure 4.4: Utility of the contract proposed at given time

cost of the other. By this means, we ensure the deal will be *mutual acceptable*, an important property of the final agreement as argued in chapter 2. This also probably leads to the number of agreements being higher. Second, within this experiments, both agents make use of the same trade-off strategy. This implies that when the deal is not symmetric, the strategy will not be symmetric (i.e. lead to a comparable outcome irrespective of the agent).

As a second measure of performance, we will look at the time at which agreement is reached (if an agreement is reached at all). In the domain of e-commerce, it is often desirable to reach an agreement as soon as possible (e.g. to reduce the communication load or because the service is required urgently). We will define this by the number of offers and counter-offers before an agreement is reached.

In undertaking this experiment, we expect that more efficient trade-off offers will lead to a more efficient eventual negotiation outcome. In the single-offer experiments we showed that by having more knowledge, the performance increased and we expect this to also be the case for the multi-offer situation. Furthermore, since the single offers are more efficient, we expect an agreement to be reached earlier:

HYPOTHESIS 6. *The more knowledge that is used in the KDE, the higher the product of the utilities will be over a complete negotiation.*

HYPOTHESIS 7. *The more knowledge is used in the KDE, the lower the communication load will be.*

First, we look at the utility. In figure 4.4, the Pareto-optimal solution which maximizes the product of utilities is indicated by the dash-dotted line. As expected, the strategy performs best when we assume full knowledge of the opponent, and worst when we do not use any knowledge at all (the *no sim* scenario). The performance of *strat100* is not significantly better than the

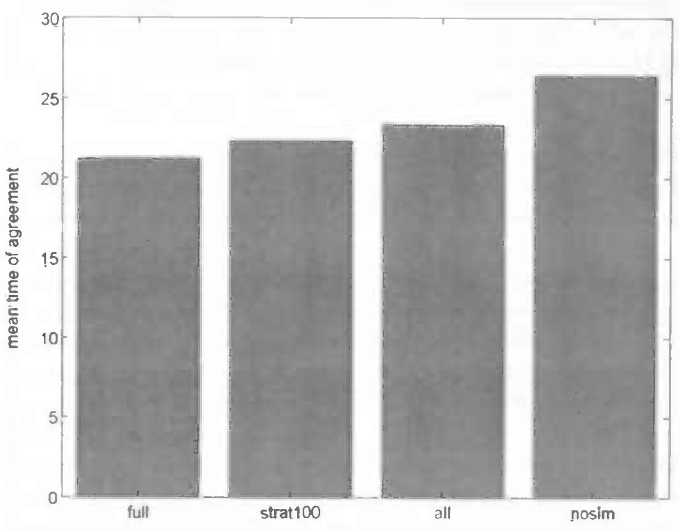


Figure 4.5: Mean time of agreement

performance of *all*, implying that the quality of the prediction of the KDE is comparable in both situations for the trade-off algorithm. As expected, the scenario in which the buyer has full knowledge about its opponent performs significantly better than the scenarios in which a KDE is used. Most important, however, the performance of the KDE-method is far better than that of not using any knowledge at all. We therefore accept hypothesis 6.

Second, we look at the mean time on which agreement is reached in figure 4.5. First, it proved that in all cases, an agreement was reached before one of the agents reached its deadline. Thus, the trade-off strategy indeed leads to mutually acceptable agreements. More specifically, when comparing the time of agreement, the scenario in which the knowledge of the opponent is complete and precise (scenario *full*), agreement is reached fastest (mean time of 21.2 steps). *Strat100* does not perform significantly worse, indicating that the prediction of the KDE-method is precise enough for the trade-off algorithm. Slightly worse is *all* which needs significantly more steps to achieve an agreement. These results can be explained by the lower performance of the single offers (as described in section 4.2). Specifically, these individual offers have a lower utility to the opponent, therefore they are less likely to be accepted. Not using the similarity criteria performs, as expected, worst. Based on this results, we also accept hypothesis 7.

Chapter 5

Conclusion and Future work

5.1 Conclusions

In this thesis we showed that the preferences of a negotiation opponent in bilateral multi-issue negotiations can be effectively learnt by using *kernel density estimation*. Specifically, by applying it to Faratin's trade-off model, we showed that it can make negotiations more efficient (in terms of utility as well as time of agreement). By choosing kernel density estimation as the learning paradigm, we did not have to make any explicit assumptions about the relation between time, negotiation history and the opponent's preferences (as many other learning methods have to). Also, the method has reasonable computational complexity for the bounded nature of e-commerce domains.

In more detail, our experiments showed that applying kernel density estimation to a single-offer case improved the performance of the trade-off algorithm. The resulting algorithm is robust when the prediction of the weights is imprecise and is not dependent on the ordinality of the weights being kept. Moreover, the amount of knowledge used in the KDEs does not have a major influence on the performance, which means it can work effectively in competitive environments in which minimal information is made available. Furthermore, we showed that using the KDE-method outperformed the uniform weight strategy for the opponent when it has a skewed weight vector and performed at least as well when it uses a near uniform weight vector (showed in section 4.2. Finally, we showed that using the KDE-method in a complete negotiation encounter which uses the predictions extensively also leads to a better performance (as showed in section 4.2.4. Specifically, the more knowledge is used in the KDEs, the higher the utility and the lower the communication load.

When we look in more detail at the requirements, as laid out in section 1.3, we can conclude that these are all (at least partly) met. The first, that the agents are computationally bounded, we fulfilled by showing that learning the kernel density estimation can be done with a complexity of $\mathcal{O}(n \log n)$ (section 3.2). The trade-off algorithm has a complexity that is linear with respect to the number of issues (section 3.3.3).

The second requirement, bounded resources in the domain, is partly met. By construction, time is bounded. However, when other resources are bounded, it is probable the agent will adapt its strategy to a resource-dependent one (as opposed to a time-dependent strategy). However, whereas the time is publicly available and the deadline for an agent assumed to be static, the amount of resource available to the opponent might or might not be available to the agent and can be dynamic (e.g. when the resource is also used in other negotiations or activities).

By construction, the third and fourth requirement are met. By testing concrete strategies, we focussed on the process off the negotiation, not merely at the efficiency of the outcome. Furthermore, our domain tested contracts consisting of multiple issues.

Kernel density estimation is a method that does not need a priori knowledge about the domain: it learns the statistical properties of the domain without explicit assumptions since it is non-parametric (see section 3.2). Furthermore, it does not assume the opponent is completely rational, as is common within game theoretic models of bargaining. Therefore, the solution is very generic, as required by the fifth requirement.

Finally, the method should be adaptive to the environment or opponent. In principle this is possible, since kernel density estimation can learn online as well as offline. However, in the experiments we only implemented the offline learning component. Therefore, this requirement is partly accepted.

5.2 Future Work

Emerging from the previous section, we can see that the current method needs extension in two ways. First, the method should be used against other strategies than the broad range of time-dependent tactics we used here. Second, an online learning component should be implemented to test the adaptiveness of the domain. In general, the method should be applied to some real-case domains to test its usability.

Besides these two direct implications, for the future, there are two main ways in which this research can be extended. Firstly, we would like to consider the performance of our method against additional meta-strategies. In this work, we only consider an opponent that uses the *smart* meta-strategy and other meta-strategies may also be adopted in practice. Secondly, our method for predicting the relative weight of the opponent's preferences for its various negotiation issues could be applied to a variety of other negotiation models where it is important to have approximations of these values. Thus, for example, it could be used in purely competitive encounters in incomplete information settings or situations in which an agent engages in multiple concurrent negotiations in order to procure a particular service. Also, the set of issues could be made more flexible, by allowing an issue set protocol (in which issues can be added or deleted from the contract under negotiation) or contingent contracts (where the values of the issues are conditional statements with respect to posterior properties of the service, like quality). To take it another step further, it would be interesting to apply the KDE-method within a richer negotiation dialogue, for example the one proposed by McBurney et al. (2002) (as discussed in 2.2.4). This model uses an argumentation formalism with strong influences from philosophy and marketing theory, supplementing a standard negotiation protocol. When improving the knowledge of an opponent, the arguments used could be better adapted to the opponent, for example by focussing on the more important issues.

Thus, having a better knowledge of the opponent's preferences could improve the performance of these mechanisms. In general, for all cases mentioned above it will hold, the better the approximation, the more acceptable the deal will be, most importantly for the agent itself, but also for its opponent, thereby leading to more efficient negotiations.

A final interesting field would be the application of the KDE-method to real human-to-human negotiation as a decision support system, although the current system is probably too coarse for this, since it is only tested on time-dependent opponents. However, in the future this method could help with designing and deepening the knowledge on formal analysis of negotiations. That this can be of great help during real negotiations is indicated by one of the analysts working on the *Panama Canal treaty*, Ken Bleakely: "It can be used to explore alternative packages of issues and can help in the making of arguments for and against various proposed sets" (Raiffa, 1982). As Bleakely also points out, using formalisations of the assumptions and trade-offs, a certain creativity was generated: "It gets people to think about the integrative aspects of bargaining, not only the distributive ones." (Raiffa, 1982)

Appendix A

Proof of Correctness Trade-Off Algorithm

In this proof, the lines of the program are indicated by a line number. Parts of the proof are between curly brackets ({...}). Within these lines, a logical term can be defined by a letter within parentheses followed by the term (e.g. (A) : *term*. After such a definition, the term is used by this letter (e.g. (A)). The letter P is used for preconditions, Q for post-conditions, B for the guard of a loop and J for the invariant of the loop. Other terms are denoted by A followed by a number. Finally, comments on the proof or program lines are put between parentheses combined with an asterisk (e.g. $(* \dots *)$).

Inputs

y^i	: Last best contract	
E	: Step utility increase.	$\{ E > 0 \}$
$V()$	: Value scoring function.	$\{ V() : y^a \rightarrow [0,1] \}$
w	: Weight vector of i.	$\{ \sum_j w_j = 1 \}$

Outputs

y^{i+1}	: Output contract.	$\{ V(y^{i+1}) = V(y^i) + E \}$

Begin

```
0. For all $j \in J$ do
1.     $\overline{E}_j = [0, 1 - V_j(y_j)]$
2. End for
           $(* \overline{E}_j$ is the set of possible amounts of improvement on issue $i *)$
       $\{ (P)$: $E > 0 \wedge$ (A1): $\forall j \in J : \overline{E}_j = [0, 1 - V_j(y_j^i)]$ where $0 \leq V_j(y_j^i) \leq 1 \}$
       $\{ (A1) \Rightarrow \overline{E}_j \subseteq [0,1] \}$

3. $E_{\max} = \sum_j w_j \max \overline{E}_j$
           $(* E_{\max}$ is the maximal improvement possible on contract $y_j^i *)$
           $(*$ From (A1) follows $\max(\overline{E}_j) = 1 - V_j(y_j) *)$
       $\{ (P) \wedge (A1) \wedge E_{\max} = \sum_j w_j (1 - V_j(y_j^{i,t})) \}$
4. $\delta = 0.01 \cdot E_{\max}$
5. If $(E_{\max} > E + \delta)$ then
```

```
                (* $\delta$ needed to guarantee convergence if $E$ is close to $E_{\max}$ (see
                3.3) *)
            { (P) $\wedge$ (A1) $\wedge E + \delta < E_{\max} \wedge E_{\max} = \sum_j w_j(1 - V_j(y_j^{i,t}))$ }
                (* $A + \beta < C \wedge \beta \geq 0 \Rightarrow A < C$ *)
            { (P) $\wedge$ (A1) $\wedge$ (J): $E < E_{\max}$ }
                (* Thus, there exists a valid contract $x$ with the desired utility:
                Stated formally: *)
            { Conclusion : $\exists x : V(x) = V(y^t) + E \wedge (\forall j \in J : x_j \in \mathcal{D}_j)$ }
6.      $k := 0$; $E_n := 0$;
                (* Write out (P) *)
            { (A1) $\wedge$ (A2) $\wedge$ (J) $\wedge (E > 0 \wedge E_n = 0 \Rightarrow E_n < E)$ }
            { (A1) $\wedge$ (A2) $\wedge$ (J) $\wedge$ (B): $E_n < E$ }
7.      While ($E_n < E$) do
                { [...] $\wedge$ (B): $E_n < E$ }
                    (* Initialize new row in $r$-matrix to 0 *)
8.          $k := k + 1$; For all $j \in J : r_j^k = 0$ ;
9.          For all $j \in J$ do
10.             If ($E_n < E$) do
                        { [...] $\wedge$ (B) }
                            (* There is still space for improvement *)
11.                 $r_i^k = \min(\text{random}(\overline{E}_j), \frac{E - E_n}{w_j})$
                            (* (i) $\overline{E}_j$ is the set of possible improvements; hence the cho-
                            sen improvement is necessarily achievable. (ii) The second
                            term is the amount of consumption which still has to be dis-
                            tributed, scaled by the issue weight; hence the consumption
                            by this issue is maximal equal to the remaining improve-
                            ment on the contract necessary. *)
                            (* $E_n < E \wedge w_i > 0 \Rightarrow \frac{E - E_n}{w_j} > 0$ *)
                        { [...] $\wedge$ (B) $\wedge 0 \leq r_j^k < \max(\overline{E}_j) \wedge 0 < r_j^k < \frac{E - E_n}{w_i}$ }
12.                 $E_n = E_n + w_i \cdot r_j^k$
                            (* Utility improvement on issue $j$ weighted by the issue
                            weight is new achieved utility $E_n$ *)
                        { [...] $\wedge$ (B) $\wedge 0 \leq r_j^k < \max(\overline{E}_j) \wedge 0 < r_j^k < \frac{E - E_n}{w_i} \wedge E_n^{i+1} =$
                        $E_i^t + w_i \cdot r_j^k$ }
                        { [...] $\wedge$ (B) $\wedge E_n^i \leq E_n^{i+1} \leq E_n^i + w_i \cdot \frac{E - E_n}{w_i} = E$ }
                            (* There is a positive probability that $E_n$ increases, with
                            a maximum value equal to $E$, the total needed improve-
                            ment *)
13.                 $\overline{E}_j = [0, \max(\overline{E}_j) - r_j^k]$
                            (* The maximal needed value of $\overline{E}_j$ is adapted to the change
                            induced by $r_j^k$ *)
                        { [...] $\wedge$ (B) $\wedge 0 \leq r_j^k < \max(\overline{E}_j) \Rightarrow \max(\overline{E}_j) - r_j^k > 0 \wedge E_n^i \leq$
                        $E_n^{i+1} \leq E$ }
                        { (A1) $\wedge$ (A2) $\wedge$ (J) $\wedge$ (B) $\wedge E_n^i \leq E_n^{i+1} \leq E \wedge \overline{E}_j \subseteq [0,1]$ }
                            (* Hence the achieved utility has a positive probability of
                            increasing, while none of the preconditions has changed *)
14.             End if
15.         End for
16.     End while
```

```
            (* Since $E_n$ has a positive probability of increasing for every issue,
            the term $(E - E_n)/w_j$ can decrease with every issue. $E_j$ only
            decreases with issue $j$, thereby guaranteeing that finally $r_j^k := (E -
            E_n)/w_i < E_j$. *)
          { $(\neg B)$: $E_n \geq E \wedge$ (A1) $\wedge$ (A2) $\wedge$ (J) $\wedge E_n \leq E \Rightarrow E_n = E$ }
17.    For all $j \in J$ do
18.       $\overline{E}_j = \sum_{i=0}^{k} r_j^i$
                (* (P) guarantees existence of this sum. *)
                (* For all $k$: $r_j^k < \max(\overline{E}_j) - w_i \cdot r_j^{k-1}$ where $r_j^0 = 0$. *)
            { $\overline{E}_j \leq \max \overline{E}_j^0 \overset{A1}{=} (1 - V_j(y_j))$ }
                (* The total improvement on this issue is achievable *)
19.       $y_j^{i+1} = V_j^{-1}(V_j(y_j^i) + E_j)$
                (* $V_i^{-1}(V_i(\alpha)) = \alpha$ *)
            { $y_j^{i+1} = V_j^{-1}(V_j(y_j^i) + E_j) \Rightarrow V_j^{-1}(V_j(y_j^{i+1})) = V_j^{-1}(V_j(y_j^i) +
            E_j)$ }
            { $V_j(y_j^{i+1}) = V_j(y_j^i) + E_j$ }
20.    End for
21. Else
          { $E_{\max} < E + \delta \Rightarrow E_{\max} \leq E$ }
            (* So, there does not exist a valid contract fulfilling the require-
            ment of having a utility of the contract $y^t$ plus E: Again, stated
            formally: *)
          { Conclusion : $\neg(\exists x : V(x) = V(y^t) + E \wedge (\forall j \in J : x_j \in \mathcal{D}_j))$ }
22.    Raise error: No step can be performed
23. End if
End
      { $\exists x : V^i(x) = E \wedge (\forall j \in J : x_j \in \mathcal{D}_j)) \Rightarrow V_j(y_j^{i+1}) = V_j(y_j^i) + E_j$ }
      { $\neg(\exists x : V^i(x) = E \wedge (\forall j \in J : x_j \in \mathcal{D}_j)) \Rightarrow$ error }
```

■

Bibliography

Amgoud, L., Maudet, N., and Parsons, S. (2000). Modelling dialogues using argumentation. In Durfee, E., editor, *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS 2000)*, pages 31–38, Boston, Massachusettes. IEEE Press.

Binmore, K. (1990). *Essays on the Foundation of Game Theory*. Basil Blackwell, Oxford.

Binmore, K. (1992). *Fun and Games*. D.C. Heath and Company, Lexington, MA.

Binmore, K. and Dasgupta, P. (1989). *The Economics of Bargaining*. Basil Blackwell, Oxford.

Bui, H., Venkatesh, S., and Kieronska, D. (1995). An architecture for negotiating agents that learn. Technical report, Department of Computer Science, Curtin University of Technology, Perth, Australia.

Bui, H. H., Kieronska, D., and Venkatesh, S. (1996). Learning other agents' preferences in multiagent negotiation using the bayesian classifier. In Shrobe, H. and Senator, T., editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference, Vol. 2*, pages 114–119, Menlo Park, California. AAAI Press.

Cohen, P. R. (1995). *Empirical Methods for Artificial Intelligence*. The MIT Press, Cambridge, Massachusetts.

Excelente-Toledo, C. (2003). *The Dynamic Selection of Coordination Mechanisms in Multiagent Systems*. PhD thesis, School of Electronics and Computer Science, University of Southampton.

Faratin, P. (2000). *Automated Service Negotiation Between Autonomous Computational Agents*. PhD thesis, University of London, Department of Electronic Engineer Queen Mary & Westfield College.

Faratin, P., Sierra, C., and Jennings, N. (2002). Using similarity criteria to make issue tradeoffs in automated negotiations. *Artificial Intelligence*, 142(2):205–237.

Faratin, P., Sierra, C., and Jennings, N. R. (2000). Using similarity criteria to make negotiation trade-offs. In *Proc. 4th Int. Conference on Multi-Agent Systems*, pages 119–126, Boston, USA.

Fatima, S., Wooldridge, M., and Jennings, N. R. (2003a). Optimal agendas for multi-issue negotiation. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 129–136. ACM Press.

Fatima, S. S., Wooldridge, M., and Jennings, N. R. (2002). Multi-issue negotiation under time constraints. In *Proceedings of the First International Conference on Autonomous Agents and Multiagent Systems (AAMAS-02)*, pages 143–150, Bologna, Italy.

Fatima, S. S., Wooldridge, M., and Jennings, N. R. (2003b). Bargaining with incomplete information. In *Annals of Mathematics and Artificial Intelligence*. Johns Hopkins University Press.

Fisher, R., Ury, W., and Patton, B. (1991). *Getting to Yes*. Penguin Books, 2 edition.

Fudenberg, D. (1985). Infinite horizon models of bargaining with one-sided incomplete information. In Roth, A., editor, *Game Theoretic Models of Bargaining*, pages pp. 73–98. Cambridge University Press.

Gerding, E. and van Bragt, D. (2003). Multi-issue negotiation processes by evolutionary simulation, validation and social extensions. *Computational Economics*, 22(1):39–63.

He, M. and Jennings, N. (2003). Southamptontac: An adaptive autonomous trading agent. *ACM Transactions on Internet Technology*, 3(3):218–235.

He, M., Jennings, N. R., and Leung, H.-F. (2003). On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):985–1003.

Jennings, N. R., Faratin, P., Lomuscio, A. R., Parsons, S., Sierra, C., and Wooldridge, M. (2001). Automated negotiation: prospects, methods and challenges. *Int. J. of Group Decision and Negotiation*, 10(2):199–215.

Kamen, E. W. and Heck, B. S. (2000). *Fundamentals of Signals and Systems Using the Web and Matlab*. Prentice Hall, New Jersey, 2 edition.

Kraus, S. (1997). Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 94(1-2):79–97.

Lancaster, K. J. (1966). A new approach to consumer theory. *The Jounal of Political Economy*, 74(2):132–157.

Mas-Colell, A., Whinston, M., and Green, J. (1995). *Microeconomic Theory*. Oxford Univ. Press.

McBurney, P., Van Eijk, R. M., Parsons, S., and Amgoud, L. (2002). A dialogue game protocol for agent purchase negotiations. *J. Autonomous Agents and Multi-Agent Systems*. in press.

Nash, J. (1950). The bargaining problem. *Econometrica*, 18(2):155–162.

Oliver, J. R. (1997). A machine-learning approach to automated negotiation and prospects for electronic commerce. *Journal of Management Information Systems*, 13(3):83–112.

Parsons, S., Sierra, C., and Jennings, N. (1998). Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292.

Raiffa, H. (1982). *The Art and Science of Negotiation*. Harvard University press, Cambridge, Massachusetts.

Rasmussen, E. (1989). *Games and Information - An Introduction to Game Theory*. Blackwell Publishers, Cambridge, Massachusetts.

Rosenschein, J. and Zlotkin, G. (1994). *Rules of Encounter.* The MIT Press, Cambridge, Massachusetts.

Rubinstein, A. (1982). Perfect equilibrium in a bargaining model. *Econometrica*, 50(1):97–109.

Rubinstein, A. (1985). A bargaining model with incomplete information about time preferences. *Econometrica*, 53(5):1151–72.

Russell, S. and Norvig, P. (2003). *Artificial Intelligence - A modern approch.* Englewood Cliffs, NJ: Prentice-Hall (2d Edition).

Sandholm, T. and Vulkan, N. (1999). Bargaining with deadlines. In *Proceedings of 16th National Conference on Articial Intelligence (AAAI-99)*, pages 44–51.

Sandholm, T. W. (1993). An implementation of the contract net protocol based on marginal cost calculations. In *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, pages 295–308, Hidden Valley, Pennsylvania.

Sandholm, T. W. (1999). Distributed rational decision making. In Weiss, G., editor, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pages 201–258. The MIT Press, Cambridge, Massachusetts.

Shapley, L. (1953). A value for n-Person games. *Contributions to the Theory of Games II.*, pages 307–317. Kuhn, H. & Tucker, A.W. (eds).

Smith, R. G. (1988). *The contract net protocol: high-level communication and control in a distributed problem solver*, pages 357–366. Morgan Kaufmann Publishers Inc.

Soo, V.-W. and Hung, C.-A. (2002). On-line incremental learning in bilateral multi-issue negotiation. *AAMAS'02.*

Sycara, K. (1989). Multi-agent compromise via negotiation. In Gasser, L. and Huhns, M., editors, *Distributed Artificial Intelligence (Vol. 2).* Morgan Kaufmann, Los Altos, CA.

von Neumann, J. and Morgenstern, D. (1944). *Theory of Games and Economic Behavior.* Princeton University Press.

Walton, D. N. and Krabbe, E. C. W. (1995). *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning.* State University of New York Press, Albany.

Wand, M. and Jones, M. (1995). *Kernel Smoothing.* Chapman & Hall, London.

Weiss, G., editor (1999). *Multiagent Systems: a Modern Approach to Distributed Artificial Intelligence.* The MIT Press.

Wooldridge, M. (1997). Agent-based software engineering. *IEE Proceedings Software Engineering*, 144(1):26–37.

Zeng, D. and Sycara, K. (1998). Bayesian learning in negotiation. *International Journal Human–Computer Studies*, 48:125–141.