Agent Learning for Automated Bilateral Negotiations

Department of Computer Science
Royal Holloway, University of London, UK

A thesis submitted for the
degree of Doctor of Philosophy
by

Pallavi Bagga

October 2021

To My Beloved Family...

(BadeDaddyJi, BadeMummyJi,

Papaji, MummyJi, and Sunny)

Declaration of Authorship

I, Pallavi Bagga, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed:

Dated:

Abstract

The potential of automated negotiating agents is high as it plays a prominent part in various domains, such as economics, behavioural psychology, and commerce systems. However, in the literature, most of the negotiating agents use fixed or heuristic strategies which possess scalability issues as they may play well in one domain but not in another. Henceforth, endowing negotiating agents with a learning ability has gained a great deal of attention in the community of automated negotiation recently, in order to help obtain the beneficial agreement in a variety of negotiation situations. In this thesis, we explore the idea of using a Deep Reinforcement Learning (DRL) approach to develop learnable strategies for self-interested agents in the domain of automated bilateral negotiations.

There are various forms of negotiation which require a strategy. This thesis starts by looking at the strategy where an agent can learn when it negotiates with many agents concurrently, but individual negotiations take place bilaterally over only one issue, such as the price of an item. In this setting, we propose *ANEGMA*, a novel agent model that uses an existing actor-critic architecture-based DRL to estimate the agent's negotiation strategy. The strategy also benefits from supervised training from synthetic negotiation data generated by teachers' strategies, thereby decreasing the exploration time required for learning during negotiation. As a result, an automated agent has been built that can adapt to different negotiation domains without the need to be pre-programmed. Experimental results show that the learned strategy outperforms the state-of-the-art "teacher" strategies in a range of settings for

single-issue bilateral negotiation.

We further extend our approach to deal with one-to-one non-concurrent negotiations over multiple issues such as the size, color, and price of an item. In this setting, we propose an extended model, called *ANESIA*, that relies upon interpretable "strategy templates" representing negotiation tactics or heuristics with learnable parameters. *ANESIA* uses a meta-heuristic approach offline, to learn the best combination of these tactics so that they can be employed during negotiation. In addition, *ANESIA* assumes that the agent has only partial information about the preferences of the user and does not know the opponent agent's preferences. To handle user preference uncertainties, *ANESIA* uses a stochastic search to best approximate the real user preferences. Besides this, *ANESIA* also combines multi-objective optimization and multi-criteria decision-making techniques to generate (near) Pareto-optimal bids during negotiation. A revised model called *DLST-ANESIA* is also developed to learn the combination of tactics on-line, using DRL. Both models, *ANESIA* and *DLST-ANESIA* are experimentally evaluated, and the experiments show how these models increase the number of "win-win" outcomes.

Since *ANESIA* agents attempt to approximate the real preferences of both negotiating parties, there is uncertainty involved in their estimated preferences. To address this uncertainty while proposing bids to the opponent party, we further extend the model by introducing an additional fuzzy component and name the model *fuzzy-ANESIA*. This model involves a two-phase bid generation step involving the use of fuzzy-multi-objective optimization and fuzzy-multi-criteria decision-making methods. The experimental evaluation empirically shows that our proposed negotiation model outperforms the state-of-the-art agents (used in previous years' negotiation competition) in most of the settings.

On a short note, this thesis focuses on bilateral negotiations (i.e., negotiations

between two agents), in which the agents exchange offers in turns. It primarily contributes towards learning ability of a negotiating agent where concurrency control is required for one or more issues. During the negotiation, the domain is known to both the negotiating agents, but their preferences and behaviour are private information. Our negotiating agent seeks to reach ‘win-win’ outcome within various time constraints (such as a deadline or discount factor) including modelling the user as well as the preferences of opponent agents.

Acknowledgements

Since any attempt at any level can't be satisfactorily accomplished without the assistance and advice of learned people, I would like to express my deepest appreciation to all those who provided me the possibility to complete my Ph.D. thesis.

At the outset, a special gratitude I give to my primary supervisor, Professor Kostas Stathis. I have been amazingly fortunate to have a supervisor whose contribution in obtaining funding, direction, enthusiasm, patience, suggestions, insightful comments, and encouragement helped me to coordinate my thesis. One simply could not desire for a better supervisor. I would also like to express my gratitude to my second supervisor, Dr. Nicola Paoletti for his enthusiasm, constructive criticism, and technical suggestions during this research effort.

In addition, I would like to convey thanks to Department of Computer Science, Royal Holloway, University of London for funding my Ph.D. for three years. I am extremely grateful for their financial assistance.

A very special thanks goes to my colleagues Benedict Wilkins, Emanuele Uliana, Nausheen Shahid, and Joel Clarke for their invaluable feedback over the years.

I would also like to pay high regards to my beloved family in India and New Zealand (grandfather (Mr. Satish K. Bagga), grandmother (Late Smt. Kailash Rani Bagga), father (Er. Rajesh Bagga), mother (Mrs. Seema Bagga) and brother (Mr. Deepak

Bagga)) for their endless love, timely advice, and moral support throughout my studies with their best wishes. I highly appreciate their confidence and faith in me.

Moreover, I wish to express my gratitude to my close relatives in the UK and Canada, especially my elder cousin Pooja Di, for their affection, love, and understanding. I am also very thankful to my friends in the UK, especially Prem and Renu, who have contributed in one way or another to a less stressful life, and have been here during the good times and the hard times.

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture.

Last but not the least, I would like to thank God who made all things possible.

Contents

List of Figures

List of Tables

List of Acronyms

ANAC Automated Negotiating Agents Competition. 43

ANEGMA Adaptive NEGotiation model for e-MArkets. 51

ANESIA Adaptive NEgotiation model for a Self-Interested Autonomous agent. 78

CSO Cuckoo Search Optimization. 118

DDPG Deep Deterministic Policy Gradient. 32

DLST Deep Learnable Strategy Templates. 101

DRL Deep Reinforcement Learning. 6

FA Firefly Algorithm. 82

MAS Multi-Agents System. 4

MCDM Multi-Criteria Decision-Making. 9

MOO Multi-Objective Optimization. 9

NSGA-II Non-dominated Sorting Genetic Algorithm-II. 37

RL Reinforcement Learning. 5

SL Supervised Learning. 5

TFN Triangular Fuzzy Number. 39

TOPSIS Technique for Order Preference by Similarity to the Ideal Solution. 38

Chapter 1

Introduction

Negotiation is a process in which different parties with conflicting goals exchange offers in order to mutually explore the likelihoods of reaching an agreement [46, 45]. Humans negotiate many times a day [48], even without realizing that they do so. It has been said that 'However much you think negotiation is a part of your life, you're underestimating it' [140]. In fact, individuals during a negotiation spend a great deal of their time during negotiations, as they try to pursue their interests in the face of conflicting goals [35]. For example, a group of people may negotiate in an attempt to choose a restaurant to eat, decide a meeting time, exchange of goods or services, form employment contracts, or engage in diplomatic negotiations between countries to decide how to act on global warming. However, despite the frequency with which humans negotiate, it is not always easy for them to reach an agreement. Often, in human negotiations, negotiators fail to see what is a better deal, they make too large concessions, they reject an offer which was better than any other available offer, or they are forced to reach an agreement even when the agreement terms are not as good as other alternatives [140].

In order to avoid the above-mentioned human negotiation flaws, a large number of researchers are attempting to automate and optimize the negotiation process. As a result, negotiation has become an important research topic in many disciplines including Economics, Law, Applied Mathematics, Psychology and Sociology, and

Computer Science [23]. One of the promising approaches to automate the negotiation process is using software components referred to as agents. An agent (also known as software robot or soft bot [121]) is a computer system which is situated in some environment and can perform an autonomous action in that environment in order to meet its design objectives [150]. An advantage of this approach in the context of negotiation is that different strategies can be provided to agents, some of which can be learned from negotiation experience.

In general, an agent-based negotiation (also known as automated negotiation) allows autonomous agents to exchange the information in the form of offers and counteroffers. An *offer* is a complete solution if it instantiates a value for each negotiated issue and is currently preferred by an agent given its preferences, constraints, and the negotiation history of offers and counteroffers [112]. The set of all the possible bids[1] is known as *outcome space*. Based on the current available information, different agents exchange a range of possible offers during the negotiation. These ranges typically reduce to the final *agreement* (i.e., successful negotiation), or if they become empty, a deal is not possible (i.e., unsuccessful negotiation) [112]. In other words, when an offer is accepted by all the agents involved in the negotiation, an *agreement* (or outcome) is achieved.

Autonomous agents are usually *self-interested*, and their aim is to maximize the value of the agreement from the point of view of the human user they represent. A numerical value representing how good an outcome or agreement is for an agent is known as *utility* [150] and measures the satisfaction of an agent for a negotiation state [4]. The states with higher utility value are preferred over the states with lower utility value. For any agreement, this utility can be calculated according to the agent's *utility function*, which is based on the preferences of the user. The ultimate goal of each agent is to maximize its utility. In an automated negotiation,

[1] Throughout this thesis, we will use the terms 'bid' and 'offer' interchangeably.

it is possible for the agents to exchange tens of thousands of offers with each other before reaching an agreement. Consequently, automated negotiation is an iterative process of: 1) evaluating offers, 2) updating the available options, and 3) making counteroffers, according to the agent negotiation strategy (or high-level plan about which actions to take). The negotiator's strategy to decide which offer to send to the other opponent agent involves decision-making which directs the negotiation process and its outcomes.

In a simple negotiation, the agents may try to agree over a single issue[2] (such as a price when buying something on e-markets like E-bay), but in a more complex setting like buying a car, the agents may negotiate over a range of issues, such as price, model, colour, and mileage. The latter negotiation type makes the negotiation more difficult because the seller is unlikely to know which feature the buyer is most interested in. Likewise, the buyer is unlikely to know which type of car the seller would prefer to sell.

Although the agent's primary goal is to maximize the utility value of its agreements, in order to maximize the chances of such agreement occurring, an agent should identify the issue to be negotiated (e.g., price) that are interested to the other party. In addition to uncertainty regarding the preferences of the other party, each agent is also unaware of the behaviour of the other party. A seller may be keen to make the sale to one particular buyer, and therefore will be willing to offer a deal at a fairly low utility. Alternatively, the seller may have many other potential buyers, and is therefore keen to reach an agreement with one of them at a high value. Sometimes, the seller may not concede until the buyer concedes, or make offers in the decreasing order of preference or just make some random offers in no particular order, or just persist with the initial offer and not concede at all. The same also

[2] A negotiation over single-issue is not unrealistic for e-markets like e-Bay, where sellers advertise a product with a fixed set of issues (e.g., Lenovo, 16 GB RAM, 250 GB HDD, i7 processor) and the only issue being negotiated is price.

applies to the behaviour of the buyer.

All the above-mentioned scenarios make the human negotiations a complex and a tedious task. Thus, automated negotiation has gained increasing importance by employing labour-saving and emotion-free agents in unknown and dynamic environments such as an e-marketplace to reduce time and negotiation costs [26, 27, 29, 90] needed to reach the agreements and simultaneously increase the chance of deals where both agents gain high utility (or 'win-win' deals) [14, 89]. It is important to note that single-issue negotiation is a 'win-lose' situation [81], i.e., what one party wins the other loses (e.g., seller and buyer negotiating over the price of a laptop). On the other hand, multi-issue negotiation is a 'win-win' situation, because two parties may have different preferences on the issues; and both parties may achieve better agreement on issues that are most important for them by trading off some on those not so important (e.g., seller may care about the laptop processor and RAM, but buyer may care about the hard disk and operating system while negotiating for a laptop). Thus, through negotiation on multiple issues, they may achieve agreement on what they care about the most by conceding over some less important issues.

Other potential benefits of automated negotiations also include the opportunity of finding more interesting deals by the exploration of large outcome spaces for an agreement [60, 90], ability to improve the negotiation skills of the human user [60, 87, 116], and the potential increase in negotiation usage since the human user can avoid social confrontation [24, 90].

1.1 Motivation

A large class of Multi-Agents System (MAS) applications are often developed using heuristic strategies, e.g., [105, 106, 2], which are experimentally tested and evaluated for only particular negotiation settings and hence, they are not adaptive. Also, these strategies often don't consider the feedback received from the environment during negotiation. In a real-world setting, the feedback issue with heuristic strategies can be seen in the following two ways: firstly, there are many negotiation settings, where positive feedback of an agent performing well in one domain (or against one opponent) may become negative in another domain (or against another opponent). Secondly, the state space increases exponentially with the increase in the number of issues in the domain. As a result, developing a heuristic strategy can become a challenging task.

In this thesis, we focus on considering this feedback using Reinforcement Learning (RL) as it allows agents to develop and improve the strategy from experience in terms of feedback when there is a very little prior information about the environment and other negotiating agents, and make it adaptive. Also, RL lets the agent make decisions sequentially, i.e., the action output depends on the state of the current input and the next input depends on the action output of the previous input, unlike a Supervised Learning (SL) algorithm where the action output only depends on the input state.

According to [121], learning is one of the features required for an agent to be considered rational. A *rational* agent can be defined as the one that is expected to be self-interested in order to reach an agreement, resulting in a high utility for the agent [4]. Moreover, the learning algorithms are more suitable for applications with uncertain or dynamic environments such as an e-market, where the structure of the environment changes in terms of the number of agents, resources, or agent deadlines [108]. Furthermore, as argued in [119], the agent learning is an integral part of the negotiation mechanism. Therefore, the advantages of learning in negotiation have been addressed quite early by different authors. To sum up, it is widely recognized that the ability of agents to learn from experience (where agents receive positive or negative reinforcement/feedback from the environment), adapt and modify their behaviour is of growing importance in the development of a MAS application.

Now, the question arises how to investigate the feasibility of RL approaches in making the effective negotiations by making agents to learn from their experience. We consider Deep Reinforcement Learning (DRL) methods, in particular, as they have shown great learning ability in many environments [99] against many opponents [128, 130] and in very large state spaces [99, 128, 130]. While addressing the learning feature, we are not interested in building up the learning mechanism for a single agent without taking into account the presence of other agents in the environment in which it is situated. Instead, we are concerned with a learning mechanism in which the agent will learn while interacting with the other agents and working towards its own desired goal.

As a result, the research reported in this thesis is predominantly concerned with the problem of learning a strategy for an agent in two different situations, i.e., when an agent is engaged in:

- one-to-many bilateral[3] negotiations with different multiple unknown agents (using fixed strategies) concurrently over a single issue, and
- one-to-one bilateral negotiation with a single unknown opponent (using fixed or dynamic strategy) over multiple issues under user preference uncertainty[4].

To address the above-mentioned research problem, the following concrete research questions arise:

- Which DRL algorithm an agent should employ to learn the negotiation strategy? Should the chosen algorithm work for continuous action space (e.g., predicting the value of price to offer to the opponent agent) and discrete action space (e.g., predicting whether to accept/make an offer from/to the opponent)? Can the proposed work be used for both single and multiple issues?

[3] Bilateral negotiation means only two parties negotiate with each other over the same resource.

[4] The human users express their preferences by ranking only a few representative examples instead of providing a fully specified utility function [141], thus agents are uncertain about the preferences characterising the profile of the user.

- How can the current state of the negotiation environment be represented?

- Should the agent negotiate against opponent agents with fixed or dynamic strategies during the learning process? Can we develop a generalized negotiation strategy which is domain-independent as well as opponent-independent? Can the resulting negotiation strategy be interpretable?

- How can we learn the preferences of an unknown opponent agent during the negotiation?

- How can we estimate the preferences of the user if only partial information is given to the agent before the negotiation begins?

- How can we reach Pareto-optimal[5] agreements under incomplete information of negotiating parties? How do we deal with the uncertainty in the estimated user and opponent models during the negotiation process?

- What performance measures can we use to evaluate the decision-making process?

1.2 Hypothesis, Aims, and Objectives

The hypothesis of this thesis is that it is possible *to let the agents learn a negotiation strategy from their experience in negotiation settings, varying from negotiating against different opponent agents to negotiating in different domains.*

The central aim of this thesis is *to design a learnable negotiation model using deep reinforcement learning for concurrent and non-concurrent bilateral negotiations over one or more issues.*

Given this aim and the context set up in the research questions identified in the motivation section, the concrete objectives of this work are as follows.

[5] A Pareto-optimal solution is one which can not be improved further without sacrificing other agent's utility, i.e., if there is another solution from which one of the agents can get more than from this Pareto-optimal solution, then the other agent must get less by that other solution [123, 44]. Pareto-optimal solutions lead to 'win-win' negotiation outcomes.

- To propose and develop a learning-based bilateral negotiation model which can support self-interested agents to make decisions on behalf of their human users in concurrent and non-concurrent negotiations for one or more issues.

- To propose a strategy that takes into account (a) incomplete information about the user's and opponent's preferences; (b) estimates the user model as well as generates the bids using approaches including fuzzy-based to deal with the uncertainties in the estimated preference models; (c) reaches agreements with maximum individual and joint utility; and (d) negotiates against unknown opponent agents.

- To propose the use of generalizable and interpretable negotiation strategy with learnable choice parameters to avoid the use of one-size-fits-all negotiation strategy in all the different negotiation settings.

- To explore the idea of using both SL and DRL for a negotiating agent to decide which action to take out of a discrete as well as continuous action space.

- To generate the synthetic negotiation data to be used for supervised learning to avoid the exploration time during the DRL process.

- To investigate the use of meta-heuristic approaches for user modelling as well as estimation of (near) Pareto-optimal bids during negotiation.

- To compare the performance of proposed learnable strategy with the existing state-of-the-art negotiation strategies using different evaluation parameters such as average negotiation time, average number of negotiation rounds, average individual utility rate, average social welfare utility rate, average distance to Pareto curve, and the percentage of successful negotiations.

1.3 Contributions of the Research

The contributions made in this research work are the following:

- We propose three different variants of DRL-based agent negotiation model for automated bilateral negotiations, which can be possibly concurrent, over one or more issues.

- We propose the use of stochastic search-based approach for user preference estimation during the negotiation.

- We also propose the use of a combination of Multi-Objective Optimization (MOO) algorithm and Multi-Criteria Decision-Making (MCDM) methods to generate (near) Pareto-optimal bids.

- We explore the use of fuzzy-based MOO and MCDM approaches to address the uncertainties in the estimated user and opponent models.

- We also introduce the use of "strategy templates" to learn the best combination of acceptance and bidding tactics at any negotiation phase.

- We extend an existing state-of-the-art simulation environment to generate data and perform experiments that support agent learning for concurrent bilateral negotiation.

- We run extensive experiments on two different simulation environments for concurrent one-to-many single-issue as well as non-concurrent one-to-one multiple-issues bilateral negotiations.

In particular, we contribute mainly towards the following important goals of automated negotiation:

- *agent learn-ability* (which enhances the autonomy of an agent),

- *agent adaptiveness* (which allows agent to negotiate against variety of opponent agents)

- *concurrent negotiations* (which allows an agent to negotiate with different multiple agents at the same time),

- *single or multiple issues* (which depends on the choice of negotiation domain),
- *user preference modelling* (which allows the agent estimate the partial preferences of human users which are submitted to the agent before the negotiation begins),
- *social-welfare utility* (which leads to more 'win-win' negotiation situations based on Pareto optimality).

1.4 Structure of the Thesis

The rest of the thesis is organized as follows:

- Chapter 2 introduces the context and the background of agent-based negotiations and further motivates the need of learning in automated negotiations. This chapter also discusses the existing studies related to the work presented in this thesis and identifies the gaps in the existing state-of-the-art of learning-based negotiation literature.
- Chapter 3 presents and evaluates our first proposed DRL-based negotiation model, called *ANEGMA*, for one-to-many single-issue negotiations. The model is applied in concurrent bilateral negotiation and shows how it outperforms the current state-of-the-art.
- Chapter 4 presents a new negotiation model, called *ANESIA*, for one-to-one multiple-issues bilateral negotiation. The model introduces the notion of "strategy templates" which represent the tactics the agent should employ during the negotiation. More specifically, the model learns template choice parameters to decide which tactic to employ for accepting an offer or generating a new bid against various different opponents and considering user preference uncertainty. Here, the tactic choice parameters are learned only once (i.e., during training) and used in all the different negotiation settings (i.e., during testing).
- Chapter 5 revises the concept of learning tactic choice parameters for "strategy templates" presented in the previous chapter. We explore the use of DRL ap-

proach to estimate the choice parameter values for different tactics which helps in accumulating the learning experience from different domains and against different opponents, unlike the one-size-fits-all strategy parameters learned in the previous chapter.

- Chapter 6 presents an extension of ANESIA model called *fuzzy-ANESIA* (or *f-ANESIA*) which handles the uncertainties in the estimated user and opponent preference models by proposing a two-phase process of generating the near Pareto-optimal bids.

- Chapter 7 concludes this thesis by showing how the research goals have been met and also provides a future research road-map.

1.5 Previous Publications

Some parts of this thesis have been published in conferences and journals papers as follows:

- P. Bagga, N. Paoletti, B. Alrayes, K. Stathis. (2020) '*Deep Reinforcement Learning Approach to Concurrent Bilateral Negotiation*'. Published in the proceedings of the *29th International Joint Conference on Artificial Intelligence (IJCAI 2020), Yokohama, Japan.*

  This paper [16] includes a part of work described in Chapter 3. However, Chapter 3 discusses the existing research-related literature in detail and provides more experimental results and discussions than [16].

- P. Bagga, N. Paoletti, B. Alrayes, K. Stathis. (2021) '*ANEGMA: an Automated NEGotiation model for e-MArkets*'. Published in the *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS).*

  Chapter 3 is based entirely on this paper [15], but here the work of [15] is put in the context of the thesis. The work of [15] generalizes the work of [16] with extra experiments and evaluation.

- P. Bagga, N. Paoletti, K. Stathis. (2022) Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation. Accepted in the proceedings of the *21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022).*

  Chapter 5 is based entirely on this recently accepted paper. But here, the work of this paper is put in the context of the thesis.

- P. Bagga, N. Paoletti, K. Stathis. (2021) '*Pareto Bid Estimation for Multi-Issue Bilateral Negotiation under User Preference Uncertainty*'. Published in the proceedings of *IEEE CIS International Conference on Fuzzy Systems (Fuzz-IEEE 2021)*, Luxembourg.

  Chapter 6 is based entirely on this paper [18]. But here, the work of [18] is put in the context of the thesis.

Chapter 2

Background and Related Work

In this chapter, we present the background concepts which provide the foundations of automated negotiation. We also present the existing related work in the area of learning-based negotiation, which is the focus of the thesis. In particular, we make an attempt to identify the advantages and disadvantages of the way learning has been applied in single-issue and multi-issue bilateral negotiation. We have structured the chapter as follows. First, we discuss the preliminaries of automated negotiation in Section 2.1. Then, in Section 2.2, we describe (a) the need of learning in negotiation, (b) the different forms of learning, which allows an agent to learn a strategy in a negotiation setting, (c) the existing learning-based negotiation literature, and (d) the learning mechanisms used in this thesis along with other relevant decision-making mechanisms. This discussion is followed by providing, in Section 2.3, a brief overview of two negotiation simulation platforms used in the thesis. We summarize the chapter in Section 2.4.

2.1 Automated Negotiation

In this section, we provide a classification of the automated negotiation and in this context, we introduce the key concepts of agent technology for representing negotiation parties as agents. This discussion is followed by an explanation of what is the negotiation environment and the settings that characterize it. The section concludes with a presentation of negotiation strategies and their significance.

2.1.1 Classification of Automated Negotiation

Automated negotiation is mainly classified into two categories based on the number of parties engaged in the negotiation and interactions that occur between them [91]: *bilateral* (where only two parties negotiate with each other over the same resource), and *multilateral* (which is a general case - over n parties). In this thesis, we focus only on the former, as we are interested in the negotiations which involve private exchange of bids. We also consider *concurrent* bilateral negotiations, which involve many one-to-one bilateral negotiations happening at the same time.

The bilateral automated negotiation can be further classified as *single-issue* and *multi-issue.* We focus on both of them. We consider *single-issue bilateral negotiations* as we are interested in e-markets like E-bay where a buyer can negotiate for a fixed set of issues (such as a laptop with the configuration: 8 GB RAM, 500 GB hard disk, Lenovo model) over only one issue such as price of a laptop. On the other hand, we consider *multi-issue bilateral negotiations* as we are also interested in domains where participants can negotiate over a range of issues to reach an agreement, e.g., deciding what laptop to buy by negotiating over issues such as RAM, hard disk, model, and price.

In order to increase the effectiveness of automated negotiation applications in business and organizational activities, we need to get as close as possible to how negotiations are conducted in the real world, where parties negotiate over various issues and seek to reach an agreement maximizing their own interests. However, a multi-issue negotiation is more complex and challenging than a single-issue negotiation because of the following main reasons [81]:

- The agent preferences over multiple issues can be complex. Traditionally, agent preferences can be characterized mathematically with a utility function, such that agents make their decisions based on this function. However, it is not trivial for a human to construct such a utility function over multiple issues. It is more cumbersome when preference over one issue impacts the values of other issues; as a result, preference elicitation may take a long time or sometimes be difficult.

- The outcome space is n-dimensional (where n is a number of issues and $n > 1$). Every time an agent plans to concede, it needs to first decide the direction of concession, i.e., to concede on which issue, either issue 1 or issue 2 and so on or different combinations of these issues. Specifically, the decision on the concession direction may also depend on the opponent's preference because conceding on the issue more important to the opponent can make the offer more acceptable (i.e., 'win-win' outcome). Also, to decide how much to concede is now more complicated because the direction can impact the amount as well. Hence, the burden of computation and reasoning for the negotiation strategy is higher in a multi-issue negotiation than in a single-issue negotiation.

- Also, an agent should realize that reaching a Pareto-optimal[1] agreement is more efficient than "leaving money on the table"[2] in case of multi-issue negotiation.

There are a number of different procedures an agent can use while negotiating over multiple issues, which greatly affects the outcome. In general, these procedures specify how the issues will be settled [44]. Three main such procedures are the following:

- *package deal procedure*: in which all the issues are bundled and discussed/settled together;

- *simultaneous procedure*: in which the issues are discussed/settled simultaneously but independently of each other;

[1] The better the two agents know each other, the more likely that they can make an agreement which is Pareto-optimal [5].

[2] It means negotiators don't get as much utility as they could, as they refrain from taking the utmost advantage of another better deal [140].

- *sequential procedure*: in which the issues are discussed/settled one after another (also known as *issue-by-issue* negotiation).

In addition to maximizing an agent's individual utility, one of our concerns is ensuring *Pareto optimality* of the solutions. For this, we chose the *package deal procedure* because unlike the other two it generates Pareto-optimal outcomes, even if it is computationally more complex. It also gives rise to the possibility of making trade-offs across issues [44].

Multi-issue negotiation can be further divided into *sequential* negotiation (where each issue is negotiated in turn) as well as *integrative* negotiation (where all issues are negotiated together). We focus on the integrative negotiation, where issues are *indivisible* and *independent* of each other. Moreover, we consider *non-mediated* negotiation, where there is no middle party between the two negotiating parties. Furthermore, based on the degree of self-interest, the negotiating parties can be classified as *self-interested*, *cooperative* and *competitive*. If the agents in the negotiation always try to maximize their own utility, then they are considered to be *self-interested agents*. If the agents are maximizing their utility at the cost of their opponent's, then they are called *competitive*. Finally, if the agents try to cooperate with each other and maximize the utility keeping in mind the benefits of their opponents, then they are considered as *cooperative* agents. We assume *self-interested agents* in our work.

2.1.2 Representing a Negotiation Party as an Agent

In this thesis, we view '*an agent as a computer system which is situated in some environment and that is capable of performing autonomous action in this environment in order to meet its design objectives*' [150]. In this context, agents are expected to interact flexibly in the environment. *Flexibility* is understood in terms of reactivity, pro-activity and social ability. *Reactivity* means that the agent should be able to perceive the environment and respond to it in a timely manner, *pro-activeness*

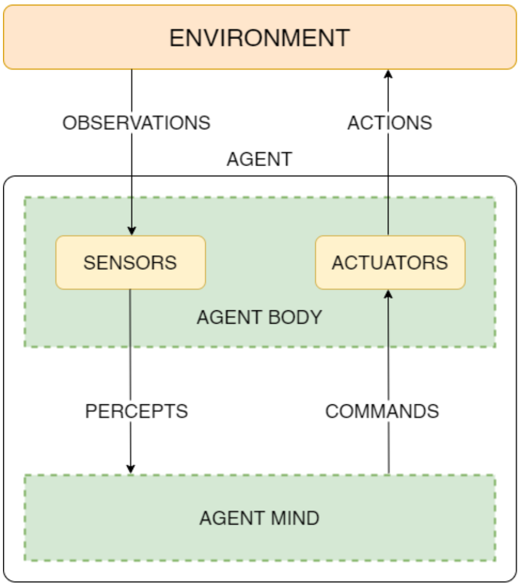

Figure 2.1: A general architecture of an agent

entails that the agent should be able to take initiative to achieve its goals, and *sociability* requires that the agent should interact with other agents or humans in the environment. Other important features of agents include: *autonomy*: meaning the agent should be able to act without the intervention of humans, *learn-ability*: implying that the agent is capable of learning or improving its knowledge from its experience with other agents in the environment and *situatedness*: signifying that the agent should be able to perceive inputs from the environment and take actions that changes the environment in some way.

Developing an agent system requires a general reference architecture, which is shown in Figure 2.1. The agent is situated in an environment [150] (such as the World Wide Web or the real world), and interacts with it through sensors and actuators [121]. The agent body aggregates the sensors and actuators in a single component, and the agent mind encapsulates the decision-making components of the agent. In other words, the agent's mind decides what action to be taken based on what it perceives from the environment through its sensors, and are executed using the actuators.

In this thesis, the environment is where agents can negotiate, e.g, an e-marketplace such as E-bay. Here, we assume that the negotiation environment is *fully-observable* for an agent, meaning that the agent has access to all aspects that are relevant for choosing an action, for e.g., a buyer agent knowing the number of seller agents in the e-market, what was the last offer made by any of the seller agents, whether the seller has accepted it s offer or not, etc. In addition, we assume that the environment is *non-deterministic*, referring to an environment in which the same action if performed twice doesn't result in the same outcome, for e.g., at one time, the seller agent already left the negotiation before it received the offer from buyer agent, resulting in failed negotiation, whereas at other time, the seller agent accepted the offer from the buyer agent, resulting in successful negotiation. In other words, uncertainty is involved. Also, we assume that the environment is *dynamic*, referring to the environment which changes over time while the agent deliberates, for e.g., in an e-market like E-bay, seller or buyer agents may enter or exit at any point of time. Moreover, we assume a *multi-agent* environment, referring to an environment with more than one agent, for e.g. a buyer negotiating with multiple sellers over price of a laptop . More details about these properties of the environment can also be found in [121].

Agents can be classified into various types, based on their capabilities and level of intelligence [121]. In this thesis, we are more interested in agents that are capable to learn from experience, and known as *learning agents*. In such agents, the mind can be thought of as being composed of four main components as shown in Figure 2.2, where agent mind and agent body are separated as in Figure 2.1. *Performance element:* selects what action to perform and send it to effectors to take action in the environment. Later, we will see that this performance element behaves like an *actor* in actor-critic RL. *Critic:* determines how well the agent is doing in the environment and give its feedback to the learning element. Later, we will see that this feedback is treated like a reward value from the environment. *Learning element:* receives feedback/reward value from critic and modifies the performance

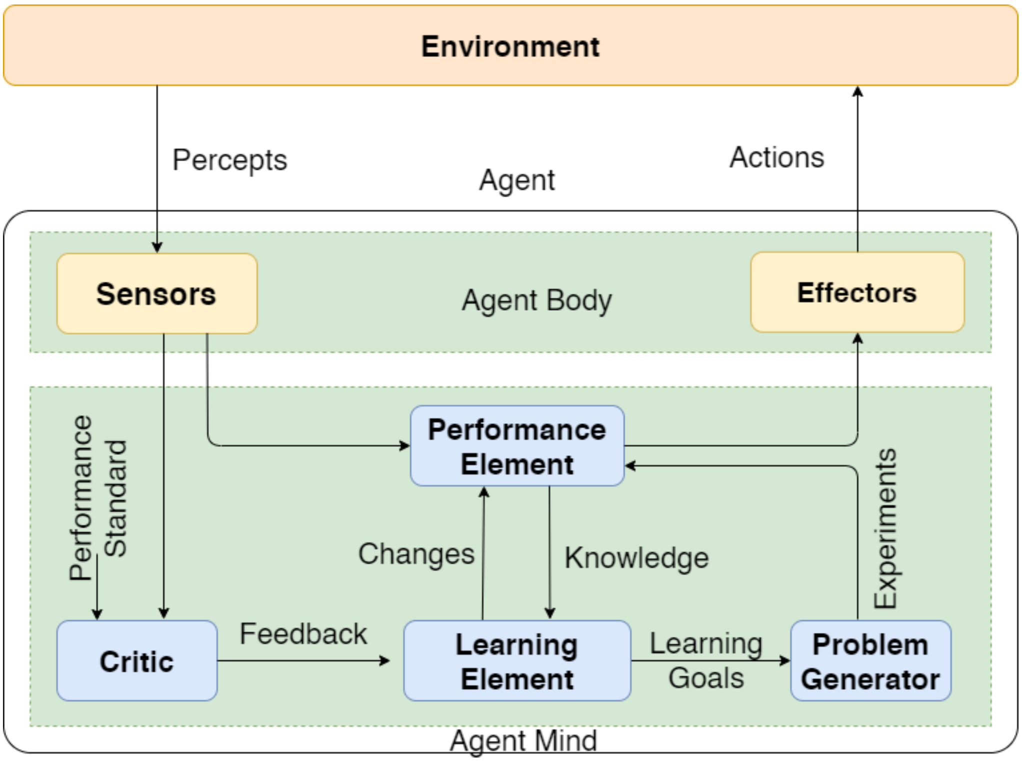

Figure 2.2: A general architecture of a learning agent, adapted from [121]

element or actor. Later, we will see that this learning element is treated like a critic in actor-critic architecture that we use in our work. *Problem Generator:* suggests actions that can lead to new and informative experiences.

2.1.3 Negotiation Environment and Settings

We assume two agents negotiating bilaterally over some domain. A negotiation *domain* is a set of one or more issues (e.g., price, colour, delivery time etc.) over which the agents negotiate to reach an agreement. Each issue can have a set of discrete (e.g., colour) or continuous (integer or real) values (e.g., price). The mapping of each issue to its value is known as a *bid* or an *outcome*. The set of all possible bids is called the *outcome space* or *negotiation space*. The total number of all possible bids in a negotiation domain is called the *domain size*. The bid/outcome which is accepted by all the parties in a negotiation is called an *agreement*.

Negotiation Protocol

Before the agents can begin the negotiation and exchange bids, they must agree on a *negotiation protocol*, which is a set of rules (like a game stating the constraints on actions taken by the agents [47]) that govern the interaction between agents, for example, which agent can participate, what are the different states of the negotiation process, what events can cause the negotiation states to change and what are the valid actions of the participants in each particular state. The negotiation protocol is known to be public, i.e., both the parties know the protocol before they start the negotiation. A comprehensive negotiation protocol for concurrent bilateral negotiations is described in [3] whereas a well-established protocol for non-concurrent bilateral negotiations is the Alternating Offers Protocol [120]. Both the protocols are discussed in detail in Chapters 3 and 4 respectively.

Preference Profile

Each agent is associated with a preference profile which describes how bids are preferred over other bids [88]. It is usually considered as private information. This is because negotiators are always unwilling to reveal their private information (e.g., parameters such as the deadline, strategies, reservation prices) to their opponents in case of being forced to a worse outcome, thus making learning in negotiation a challenging problem. When the preference profiles of agents are combined with a negotiation domain and a negotiation protocol, a *negotiation scenario* is created [70]. We assume that the preferences are never changed during the course of negotiation, although dynamic preferences have also been considered [118, 114].

Utility function

A utility function is used to represent the preference profile of an agent. This function maps each possible bid in the negotiation domain to a real value in the interval $[0, 1]$ indicating its utility for the agent. The utility function can be defined in linear or non-linear variations, but in most studies, the linear additive utility has been used. We also use linear additive utility in our work since we choose negotiation domains without (preferential) dependencies between issues, i.e., the contribution of every issue to the utility is linear and doesn't depend on the value of other issues [95]. An advantage of independence between issues is that algorithms that search for a proposal with a particular utility can be implemented in a computationally efficient way. It also makes it easier for negotiation strategies to efficiently model the preferences of parties involved, as it reduces the amount of information that is to be learned by a preference learning technique [12].

Preference Uncertainty and User modelling

If a partial ordering is used as a preference profile instead of a utility function, then it is called *preference uncertainty* [88]. In particular, the available information to the agent is that bid X is preferred over bid Y for a subset of possible bids [88]. The agent's goal is to estimate the utility function that approximates the real utility function of the user for every possible bid as much as possible, which is also referred to as *user modelling* [88]. In case of uncertain user preferences, an agent may also want to elicit more information about the real utility in order to improve the user model by querying the user during negotiation against an elicitation cost or bother cost [9, 13].

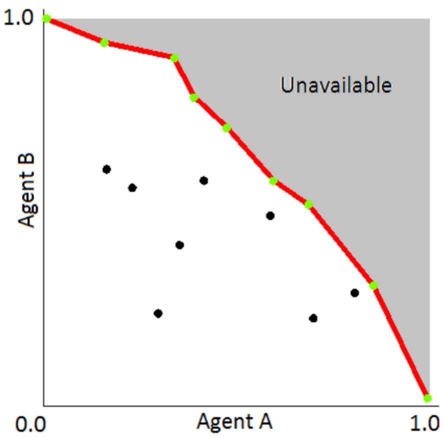

Figure 2.3: Pareto Frontier [88]

Optimality of a Bid

In general, given the set of all possible bids, there is a small subset of bids which is more preferred as outcomes by both the negotiating parties. Identifying these special bids might lead to a better agreement for both parties. From a single agent perspective, the optimal bid is the one that has maximum utility value. But, in general, an optimal bid involves the utilities of both the negotiating parties. One approach to optimality is that a bid is not optimal for both the parties if there is another bid that has the higher utility for one party and at least equal utility for the other party. This type of optimality is called *Pareto optimality.* The collection of Pareto optimal bids (green dots in Figure 2.3) is called the Pareto Frontier (red line in Figure 2.3) [88].

In multi-issue bilateral negotiation, we approach the optimality of a bid as a constraint optimization problem, where the space of agreement is not empty (see Zone of Agreement concept later in Chapter 3). In this case, at least one Pareto-optimal bid always exists. For smaller domains or discrete domains of size $|d|$, one can use Brute-force, which takes d^2 comparisons to find the Pareto-front. For larger domains or continuous domains, one can use the meta-heuristic approach as used in this thesis. We use this approach irrespective of the type (linear or non-linear) of the utility function, as this is more general, and therefore our model can be applicable in more domains, even if sometimes it will be computationally more complex unnecessarily. This is the price that we choose to pay for generality.

Scenario parameters

A negotiation scenario includes three key elements that illustrate the complexity of the negotiation process. These elements are *negotiation deadline*, *reservation value*, and *discount factor.* The *deadline* of a negotiation denotes the point of time before which an agreement must be reached. The deadline may be specified as the maximum number of negotiation rounds, or alternatively as a real-time target.

Usually, the negotiation time is normalized in $[0, 1]$ so that 0 represents the start of the negotiation and 1 represents the negotiation deadline. The *reservation value* is the lowest level of utility that the agent can accept from its opponents, and any bid with utility below that level is not acceptable. We also assume that if the negotiating agents can't reach an agreement, they will receive a utility equal to their reservation value (when dealing with multiple-issues). The *discount factor* is a way of modelling the time pressure on agents for decision-making indicating that the resource is worth less as time passes, and makes the acceptance or rejection of a bid to be more challenging. This makes the utility of bids decrease for the agents over time.

2.1.4 Negotiation Strategies

The success of a negotiating agent is determined by the effectiveness of its decision-making model. The basis of each decision-making apparatus is the *strategy* which is employed to act in line with the negotiation protocol in order to achieve the agent's objectives. In simpler terms, it states how should agents negotiate. The negotiation strategy is known to be private to each party, i.e., each party has its own strategy. This is unlike game theoretic approaches, where each party is aware of its opponent's strategies [22].

In [11], a component-based architecture for agent negotiation called BOA is proposed where a strategy is divided into three distinct components: a Bidding strategy (B), an opponent modelling (O), and an Acceptance strategy (A). The *Bidding strategy* determines a concession behaviour during negotiation and how to generate appropriate bids according to this behaviour. The *Opponent model*[3] tries to model the opponent's preference profile or behaviour style using learning techniques so that the

[3] In some negotiation problems, one of the parties may know something of relevance that the other does not. For instance, when negotiating over the price of a second hand car, the seller knows its quality, but the buyer does not. Such situations are said to have *asymmetry* of information between the parties. On the other hand, in *symmetric* information situations, both parties have the same information [44]. In this thesis, we consider the latter case.

agent can make more informed decisions and can act accordingly to cooperate with the teammates more effectively or take the best advantage of the opponents. The *Acceptance Strategy* determines whether the agent should accept the offer received from the opponent. Let us assume that an agent A_u has to decide an action $a_t \in A$ where A is a set of possible actions at time t, as a response to the opponent agent A_o's offer at time t. Since we assume that all the negotiating agents are situated in the negotiation environment, the opponent's offer comes from the environment, which along with other environment, agent and domain (single/multi-issue) parameters contribute to the agent's internal state s_t at time t. Now, we can formulate the negotiation strategy S_{A_u} of A_u as a function f in (2.1), which maps s_t at time t to an action a_t to be taken by A_u at time t.

$$S_{A_u} = a_t = f(s_t) \tag{2.1}$$

Apart from the above, developing negotiation models that let an agent learn a strategy during negotiation normally assumes the following three-phase process [77]:

- In Phase-I (or the *pre-negotiation phase*), the negotiating agent gets prepared with details such as the settings and protocol, the negotiation parameters and number of issues, and a user preference model. It also possibly involves learning the user model from given partial information[4] as well as eliciting[5] the preferences from the user under uncertainty [13, 9].

- In Phase-II (or the *negotiation phase*), the agent is deployed to negotiate, involving offer generation and additional components such as opponent model prediction, offer evaluation and acceptance.

- Finally, in Phase-III (or the *post-negotiation phase*), the optimality of the final agreement is assessed in terms of various metrics such as average individual or

[4] Human users/negotiators do not necessarily know their own utility function explicitly.

[5] Preference elicitation is a tedious procedure to the users since they have to interact with the system repeatedly and participate in lengthy queries [13, 9]. This process may continue until the negotiation terminates.

social welfare utilities, and distance to Pareto frontier, only if an agreement is reached.

2.2 Learning in Automated Negotiation

The learning capability of agents is essential for the success of any MAS [119]. Such a capability can be viewed as the ability of agents to perform unknown tasks (which are not performed before) or known tasks (the old tasks) better as a result of changes produced by the learning process [119]. The extant literature has demonstrated that the ability to learn significantly contributes to the agent's negotiation power and ability to reach agreements faster.

2.2.1 Motivation for Learning in Negotiation

As we mention in Chapter 1, in a realistic negotiation situations, agents need to work in a dynamic environment with different beliefs, goals, preferences, and levels of knowledge. Moreover, the agents may exist in environments which may fluctuate over time. The agents may also need to choose a solution/outcome out of a pool of potential solutions/outcomes. This is more imperative when the decision has to be made in a limited time. In such negotiation settings, the agents face uncertainties due to the incomplete information about other agents or the environment. Moreover, as there are many negotiation settings, an agent that performs well in one setting may become ineffective in another. Furthermore, when considering multi-issue negotiations, the state space increases exponentially with the number of issues. To this end, the agents need to learn about other agents or adapt their local behaviour based on the environment to effectively utilize the opportunities [119]. In other words, a general learning agent could help with the above challenges by scaling the agent in both depth (by handling large outcome spaces) and breadth (by handling multiple domains and opponents).

2.2.2 Key Components of Agent Learning

Agent learning is an integral part of the negotiation mechanism [119]. The various essential elements of learning on which the negotiating agents build their inference are the following [119]:

- *Agent's Expectations*: They represent the current information of the environment internal to the agent, which guides the agent's decision-making. It includes what and how much agent expects to get from the others or what the others will do. A continuous learning process will make the agent to modify its beliefs to be more realistic during the negotiation.
- *Feedback*: This may originate from direct or indirect communication with other agents, or without communication, directly through the learning agent's observations of the effects of its decisions and other agent's actions.
- *Evaluation Criteria*: They define how the agent evaluates the feedback from others as a response to the agent's last decisions or actions.

2.2.3 Different Forms of Learning

According to [147], three different forms of machine learning are normally considered according to the *learning feedback* system - in our case the negotiating agent:

- *Supervised learning*: It is a type of machine learning, in which feedback specifies the desired activity of the learning agent. The objective of the learning is to match this desired activity as much as possible.
- *Unsupervised learning*: It is a type of machine learning in which no explicit feedback is provided. The objective of learning is to find out the useful and desired activities based on trial and error and self-organized processes.
- *Reinforcement Learning*: It is a type of machine learning in which an agent learns in an interactive environment by trial and error using feedback or reward from its own actions and experiences. The reward only specifies the utility of the actual

activity of the agent. The main objective of learning is to maximize the agent's utility.

In this work, we will use Supervised Learning for training an agent to interact with the environment and Reinforcement Learning for the agent to build up experience while interacting with the environment. We explain next Reinforcement Learning, as it plays a significant role in this thesis.

2.2.4 Reinforcement Learning

Reinforcement Learning (RL) is a goal-oriented optimization technology that has shown great promise in many complex domains. It learns a mapping from states to actions, called a *policy*, to control the behaviour of an agent [136]. To obtain this policy, an agent repeatedly interacts with an environment using a trial-and-error method [136]. In general, this model consists of an agent, a set of possible states S and a set of possible actions per state A, an unknown transition function, and an unknown real-valued reward function. At each point in time, when an agent performs an action a_t, it moves from one state s_t to a new state $s_{t+1} = \delta(s_t, a_t)$ observed by the agent, and receives a reward $r_{t+1} = r(s_t, a_t)$ as a result of the executed action [145]. The interaction of the agent with the environment is shown in Figure 2.4. RL can be further classified based on model support as well as type of learning:

- *Model-based/Model-free RL*: Model-based RL has an agent try to understand the world and create a model to represent it. On the other hand, model-free RL lets an agent learn a policy directly using algorithms without learning a model [136]. In other words, if, after learning, the agent can make predictions about what the next state and reward will be before it takes each action, it's a model-based RL algorithm; otherwise, model-free. Here, model means a function which predicts state transitions and rewards. Model-free methods are easier to implement and tune as compared to model-based, this is because in the latter case, the ground-truth model is usually not available to the agent and if an agent wants to use a

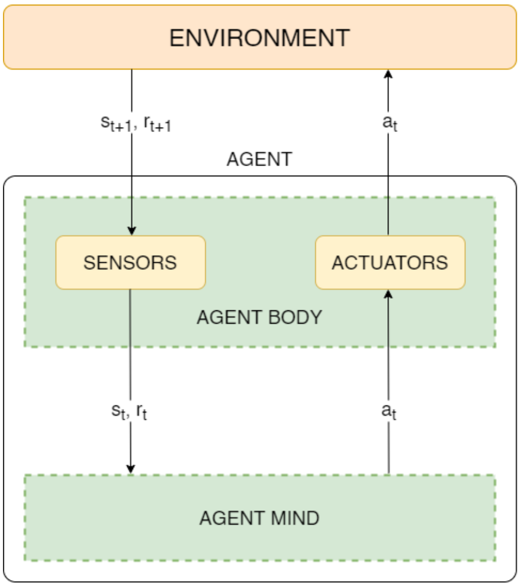

Figure 2.4: Interaction of agent with the environment during Q-Learning

model, it has to learn the model purely from experience, which creates several challenges, for e.g., exploitation of bias in the model by an agent, which results in an agent that behaves well in the learned model rather than the real environment [102] (also known as overfitting in machine learning algorithms).

- *Off-policy/On-policy RL*: On-policy methods (such as SARSA [136]) attempt to evaluate or improve the same policy that is used to make decisions, whereas off-policy methods (such as Q-learning [145]) evaluate or improve a policy different from the one that is used to make decision or select action. In other words, in an off-policy RL, as an agent learns another policy than the one it uses to select actions, the agent can continue exploration with trial-and-error actions while learning an optimal policy. However, on-policy learns sub-optimal policy. Since, off-policy allows parallel learning, learning is fast.

- *Value-based/Policy-based/Actor-Critic RL*: Value-based RL (such as Q-learning [145]) learns the state or state-action value, and then policy infers from there, i.e., agent chooses the action with the maximum value. On the other hand, policy-based (such as REINFORCE [135]) directly learns the policy function that maps state to action without evaluating the value-function. Actor-critic RL is a combination of both value-based and policy-based RL. The ‘critic’ estimates the value function (action-value: Q-value or state-value: V-value). The ‘actor’ updates the policy distribution in the direction suggested by the Critic (such as with policy gradients) [136].

Q-learning

Q-learning is one of the most popular methods to support the RL paradigm. It is a model-free, off-policy, value-based RL algorithm that resorts to observable state transitions and their corresponding rewards to estimate the long-term value of choosing an action at a given state and following the optimal policy afterwards. Although Q-learning was first introduced to address problems in single-agent environments, it could also be used in MASs [28], with quite high chance of converging to the optimal policy [139]. In general, it assigns a value to <state, action> pairs and thus, implicitly represents the policies [146]. Primarily, the goal of the agent is to find an optimal policy $\pi^* : S \rightarrow A$ that maximizes the sum of the immediate reward and the value of the immediate successor state (see (2.2)).

$$\pi^*(s) = \arg\max_a (Q^\pi(s,a)) \tag{2.2}$$

The Q-function for policy is defined in (2.3), where, $V^\pi(s)$ is a utility value based on the rewards received starting from state s and following the policy π, whereas γ is a discount factor with the range of 0 to 1 ($0 \leq \gamma < 1$) determining how much importance should be given to the future rewards. In other words, if γ is closer to 0, the agent will tend to consider only immediate rewards, whereas, if γ is closer to 1, the agent will consider future rewards with greater weight, willing to delay the reward.

$$Q^\pi(s,a) = r(s,a) + \gamma V^\pi(\delta(s,a)) \tag{2.3}$$

2.2.5 Learning-Based Negotiation Literature Review

Automated negotiation has been at the forefront of the research interests in MASs and AI communities for many years. Over time a large number of negotiation strategies have been proposed [87, 96] including: bayesian learning, constraint based learning, probabilistic decision theory, case-based reasoning, heuristic strategies, RL and evolutionary strategies. As a result, the existing automated negotiation literature is extensive. Since, our work is at the intersection of the domains of autonomous negotiation and learning agents, in this section, we discuss the existing literature on learning-based bilateral negotiation. Also, we consider Pareto-optimal solutions and user preference uncertainties in our multi-issue negotiation work, which motivates us to bring multi-objective and single-objective optimization approaches as part of thesis background. Later, we also identify the gaps in the existing state-of-the-art literature, with particular emphasis on negotiation strategies. We classify the learning-based literature using the following attributes as demonstrated in Table 2.1:

- *multilateral*, whether the negotiation occurs between more than two agents (here, *Not Tested* indicates that authors claim to support this feature, but do not test it);

- *continuous action space*, whether the action space is continuous or discrete (we put a hyphen where we could not evince this from the paper alone);

- *concurrent negotiations*, whether agents negotiate concurrently with multiple other agents;

- *dynamic environment*, whether the environment can change during the negotiation, for e.g., when new/old agents can enter/exit the environment at any time (here, (hyphen is meant as per above);

- *incomplete information*, whether negotiating parties are unaware of each other's preferences (here, (hyphen is meant as per above);

- *human-agent negotiation*, as opposed to agent-agent negotiation;

- *adaptive*, whether the model can adapt well to never-before-seen negotiation settings (either domains or opponent agents) (*Not Tested* is meant as per above);
- *mediated*, whether the negotiation between two agents involve any third party;
- *use of RL*, such as Q-learning;
- *use of DRL*, as opposed to RL approaches that do not rely on deep learning;
- *domain-independent*, whether the strategy can work for more than one domain (*Not Tested* and hyphen are meant as per above);
- *multi-issue negotiation*, as opposed to negotiating over a single issue (*Not Tested* is meant as per above);
- *Pareto-optimality*, whether the Pareto-optimal bids have been considered during experiments in multi-issue negotiations; and
- *strategy component*, whether the authors have focused on a specific component of the negotiation strategy, e.g., bidding, opponent modelling, acceptance strategy, or user modelling (in case of user preference uncertainty, including preference elicitation).

From Table 2.1, it has been observed that most of the existing negotiation approaches with RL have mainly focused on methods such as Tabular Q-learning for bidding [20] or DQN for bid acceptance [117]. However, these approaches are neither optimal for continuous action spaces nor can handle user preference uncertainty. It is also observed that (a) very few strategies have targeted the Pareto-optimal agreements; (b) few works have considered user preference modelling; (c) the use of DRL is minimal; and (d) almost no work has been done on concurrent multiple-issue negotiations. In our work, we focus on (a) to (c) and leave (d) for future work.

We choose to use the actor-critic architecture combined with DRL as they provide a rich class of strategy functions to capture the complex decision-making behind negotiation. In our research, the agent negotiates with fixed-but-unknown opponent strategies in a negotiation environment, which can be learnt by the buyers after some simulation runs. Hence, we consider our negotiation environment as *fully-observable.* Following this, for our *dynamic* environment, we use a *model-free*, *off-policy* RL approach which generates a *deterministic policy* based on the *policy gradient* method to support continuous control. More specifically, we use Deep Deterministic Policy Gradient (DDPG) algorithm (will be explained in detail in Chapter 3), which is an actor-critic RL approach and generates a deterministic action selection policy for the negotiating agent [86]. We consider a *model-free* RL approach because our problem is how to make an agent decide what action to take next in a negotiation dialogue rather than predicting the new state of the environment. In other words, we are not learning a model of the environment, as the strategies of the opponents are not observable properties of the environment's state. Thus, our agent's emphasis is more on learning what action to take next and not the state transition function of the environment. We consider the *off-policy* approach (i.e., an agent attempts to evaluate or improve the policy which is different from the one which was used to take an action) for independent exploration of continuous action spaces [86].

2.2.6 Optimization for Automated Negotiation

This thesis also requires an optimization process for finding the optimal solution out of many possible solutions, such as finding which bid to offer from a set of bids, while aiming for 'win-win' solutions, or estimating the user model using only given partial preferences of the user.

Meta-heuristics for Automated Negotiation

Meta-heuristics are generally a family of approximate optimization techniques that involves an interaction between local improvement procedures (heuristics) and higher level strategies (with the use of memory, solution history and other forms of learning [152]) in order to find global optimal solutions for a problem [51, 138]. Although finding global optimal solutions is not always guaranteed, they can provide "acceptable" solutions in a reasonable time for solving complex problems by efficiently exploring the search space [138].

Based on the multiplicity of solutions that are manipulated in the search guiding process, meta-heuristics algorithms can be categorized as single-solution based and population-based methods [51, 138]. In *single-solution based algorithms* (such as Local Search [61], Simulated Annealing [76] and Tabu Search [52]), a single solution is manipulated or transformed during the search. The *population-based algorithms* (such as evolutionary algorithms, e.g., genetic algorithms [98], and swarm intelligence techniques, e.g., particle swarm optimization [75]) guide the search procedure by working on a number of solutions (based on the notion of population) sampled from the search space and evolve them through the search until they include acceptable solutions. More recently, many nature-inspired meta-heuristic methods have also been developed [152], such as the Bat algorithm, the Firefly algorithm, and the Cuckoo search, and most such algorithms are based on swarm-intelligence [79]. Although, meta-heuristic methods are computationally-expensive [152], they are more prominent than other traditional algorithms (such as linear programming) as: (a) they often find true global optimality; (b) they can solve a wider range of problems as they often treat problems as a black-box; (c) they are usually gradient-free; and (d) they are stochastic and, hence, no identical solution can be obtained, even when starting with the same initial points.

Single-objective optimization for User Modelling

For multi-issue bilateral negotiations, we assume that each agent has its own private preference profile describing how bids are offered over the other bids in terms of a utility function U. U is defined as a weighted sum of evaluation functions $e_i(v^i_{c_i})$, as shown in (2.4).

$$U(\omega) = U(v^1_{c_1}, \dots v^n_{c_n}) = \sum_{i=1}^{n} w_i \cdot e_i(v^i_{c_i}), \text{ where } \sum_{i=1}^{n} w_i = 1 \tag{2.4}$$

In (2.4), each issue i is evaluated separately and contributes linearly to the utility U. This is a very common utility model and is also known as *Linear Additive Utility space*. Here, w_i are the normalized weights indicating the importance of each issue i to the user, while $e_i(v^i_{c_i})$ is an evaluation function that maps the $v^i_{c_i}$ value of the i^{th} issue to a utility. Here, an agent's bid ω is a mapping from each issue to a chosen value (denoted by c_i for the i-th issue), i.e., $\omega = (v^1_{c_1}, \dots v^n_{c_n})$. Note that the linear utility function does not take dependencies between issues into account.

In our settings, where the negotiation environment contains *incomplete information*, because the user utility model U_u is unknown. Only partial preferences are given for the user, i.e., a partial order $\preceq$ over B bids w.r.t. U_u s.t. $\omega_1 \preceq \omega_2 \rightarrow U_u(\omega_1) \leq U_u(\omega_2)$.

Hence, during the negotiation, one of the objectives of our agent is to derive an estimate $\widehat{U}_u$ of the real utility function U_u from the given partial preferences[6]. This leads to a single-objective constrained optimization problem, expressed as (2.5)[7]:

$$\begin{aligned} \max_{\substack{\widehat{w_1},\dots,\widehat{w_n},\\ \widehat{e_1}(v^1_{c_1}),\dots,\widehat{e_n}(v^n_{c_n})}} \quad & \rho\left(\sum_{i=1}^{n} \widehat{w_i} \cdot \widehat{e_i}(v^i_{c_i}), B_{\preceq}\right) \\ \text{s. t.} \quad & \sum_{i=1}^{n} \widehat{w_i} = 1 \\ & \widehat{w_i} > 0 \text{ and } 0 \leq \widehat{e_i}(v^i_{c_i}), \forall i \in n \end{aligned} \tag{2.5}$$

$B_{\preceq}$ is the incomplete sequence of known bid preferences (ordered by $\preceq$), and ρ is a measure of ranking similarity (e.g., Spearman correlation) between the estimated ranking of $\widehat{U}_u$ and the true, but partial, bid ranking $B_{\preceq}$.

[6] Humans do not necessarily use an explicit utility function. Also, preference elicitation can be tedious for users since they have to interact with the system repeatedly [13]. As a result, agents should accurately represent users under minimal preference information [141].

[7] We note that this problem is under-determined, i.e., there are multiple solutions for $\widehat{\omega_i}$ and $\widehat{e_i}$ to maximize the similarity.

Table 2.1: Comparison between learning-based negotiation strategies

Reference	Multilateral	Continuous Action Space	Concurrent Negotiations	Mediated	Dynamic Environment	Incomplete Information	Human-Agent Negotiation	Adaptive	Use of RL	Use of DRL	Domain-Independent	Multi-Issue Negotiation	Pareto-optimality	Strategy component			
														Bidding	Acceptance Strategy	Opponent Modelling	User Modelling
[30]	✓	✓	✗	✓	✗	–	✗	✓	✓	✗	✗	✓	✗	✓	✗	✗	✗
[133]	✗	✓	✗	✗	✗	–	✗	✗	✓	✗	Not Tested	✓	✗	✗	✗	✓	✗
[139]	✗	✓	✗	✗	✗	–	✗	–	✓	✗	✓	✓	✗	✓	✗	✗	✗
[68]	✗	✓	✗	✗	✗	✓	✗	✗	✓	✗	Not Tested	✓	✗	✗	✗	✓	✗
[134]	✗	✓	✗	✗	✗	–	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✗
[160]	✗	✓	✗	✗	✗	✓	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗
[34, 33]	✗	✓	✗	✓	✗	✓	✗	✓	✓	✗	Not Tested	✓	✗	✓	✗	✗	✗
[109]	✗	–	✗	✗	–	–	✗	Not Tested	✓	✗	Not Tested	✓	✗	✗	✗	✗	✗
[83]	✗	–	✗	✗	–	–	✗	Not Tested	✓	✗	Not Tested	✓	✓	✗	✗	✗	✗
[20]	✗	✗	✗	✗	–	✓	✗	Not Tested	✓	✗	✓	✓	✓	✓	✓	✓	✗
[65]	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✓	✓	✗
[117]	✗	✗	✗	✗	✗	–	✗	Not Tested	✓	✓	✓	✓	✓	✗	✓	✗	✗
[31]	✗	✓	✗	✗	–	–	✗	✓	✓	✓	Not Tested	✓	✓	✓	✓	✗	✗
[148]	✗	✗	✓	✓	–	✓	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗
[156]	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗	✓	✓	✓	✗	✗	✓	✗
[54, 55]	✗	✗	✗	✗	✗	✓	✗	✓	✓	✗	✓	✓	✗	✓	✓	✗	✗
[29]	✗	✗	✗	✗	✗	–	✗	✗	✗	✗	Not Tested	✓	✗	✗	✗	✓	✗
[80]	✗	✓	✗	✗	✗	✓	✗	✗	✓	✗	–	✓	✗	✓	✓	✗	✗
[155]	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✓	✓	✓	✗	✗	✓	✗
[59]	✗	–	✗	✗	✗	✓	✗	✗	✗	✗	✓	✓	✓	✗	✗	✓	✗
[25]	✗	–	✗	✗	✗	✓	✗	✗	✗	✗	Not Tested	✓	✗	✗	✗	✓	✗
[154]	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗	Not Tested	✗	✗	✓	✗	✓	✗
[158]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	Not Tested	✓	✗	✓	✗	✓	✗
[84]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗
[137, 157]	Not Tested	✓	✗	✗	✗	✓	✗	Not Tested	✗	✗	Not Tested	Not Tested	✗	✓	✗	✗	✗
[119]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
[159]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✗
[122]	✗	–	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
[103]	✗	–	✗	✗	✓	✓	✗	✓	✗	✗	–	–	✗	✗	✗	✓	✗
[21]	✗	–	✗	✗	✗	–	✗	Not Tested	✗	✗	✗	✓	✗	✗	✗	✓	✗
[62]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	Not Tested	✓	✗	✗	✗	✓	✗
[82]	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	–	✓	✓	✗	✗	✓	✗
[37]	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	–	✓	✗	✗	✗	✓	✗
[132, 131]	✗	✓	✗	✗	✗	✓	✗	✓	✗	✗	–	✓	✗	✗	✗	✓	✗
[11]	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗
[15, 16]	✗	✓	✓	✗	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
[18]	✗	✗	✗	✗	✗	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Many meta-heuristic optimization algorithms have been adopted by researchers in multi-issue automated negotiation, such as Particle Swarm Optimization for opponent selection [127]; and Genetic Algorithms [38], Chaotic Owl Search [41], Hill Climbing (or local search) [126, 74] and Simulated Annealing [78, 126] for exploring the outcome space in order to find the desired offers for generating bids during the negotiation. These algorithms focus on different problem areas than ours, as we are solving a constraint-satisfaction problem of estimating the user model that best agrees with the given partial preference ranking order. This is important in order to achieve optimal negotiation results given incomplete information on the human user the agent represents. In our work (see Chapters 4 to 6), we explore the idea of using nature-inspired *single-objective* (see 2.2.6) meta-heuristic algorithms called Firefly algorithm [151] and Cuckoo Search Optimization algorithm [153] for estimating the user utility model from the given partial preferences. These have been widely used in many engineering problems, but not in the domain of bilateral negotiation. Both algorithms are explained in detail in Chapters 4 and 6 respectively.

Multi-objective optimization for Pareto estimation

Originally, the idea of generating a Pareto-optimal offer with perfect information was proposed by Raiffa in [113]. Jazayeriy et al. in [66] presented the Maximum Greedy Trade-offs algorithm to generate Pareto-optimal offers with perfect information. This was further extended in [67] to generate near Pareto-optimal offers with incomplete information of the opponent's preferences, but complete preference information of the user. Sanchez-Anguix et al. in [125, 124] proposed a bottom-up approach to achieve a Pareto-optimal solution in a group decision-making setting. By assuming incomplete opponent preferences only, they also provided proof that a Pareto-optimal solution in a sub-group is also Pareto-optimal in the super-group containing the sub-group. In our work, we assume that the utility models of both user and opponent agents are not given while generating Pareto-optimal offers.

Ehtamo et al. in [40] considered a non-biased mediator-based negotiation approach assuming that agents know their utility function during the negotiation and introduced a constraint proposal method to reach the Pareto-optimal solutions, which was extended later to multi-party negotiations in [58]. Hara and Ito in [56] used a mediator-based negotiation approach in which a GA was used over interdependent multiple issues. Instead, in our proposed approach, we do not rely on mediation, and we assume independent issues and thus avoid the extra cost of a mediator during the negotiation. A mediator was also used to exclude the unreasonable (or less beneficial for buyer agent) offers from the feasible set of negotiation offers by the negotiation strategy in the work of Montazeri et al. [100]. This work generated Pareto solutions with the help of DRL for e-commerce, considering only preference information about the opponent.

We have already seen that in our negotiation settings, we assume that each agent has incomplete information of other agent's preferences and behaviour. To increase the agreement rate over multiple issues, an agent must estimate the preferences of other agents to generate the (near) Pareto-optimal bid. This is a multi-objective optimization (MOO) problem and can be defined as follows:

$$\max_{\omega \in \Omega} \quad (\widehat{U}_u(\omega), \widehat{U}_o(\omega)) \tag{2.6}$$

In (2.6), we have two objectives: $\widehat{U}_u$, the user's estimated utility, and $\widehat{U}_o$, the opponent's estimated utility. A bid $\omega^* \in \Omega$ is Pareto-optimal if no other bid exists $\omega \in \Omega$ that Pareto-dominates ω^*. In our case, a bid ω_1 Pareto-dominates ω_2 iff:

$$\begin{gathered}\left(\widehat{U}_u(\omega_1) \geq \widehat{U}_u(\omega_2) \land \widehat{U}_o(\omega_1) \geq \widehat{U}_o(\omega_2)\right) \land \\ \left(\widehat{U}_u(\omega_1) > \widehat{U}_u(\omega_2) \lor \widehat{U}_o(\omega_1) > \widehat{U}_o(\omega_2)\right)\end{gathered} \tag{2.7}$$

In order to find Pareto-optimal solutions, the Genetic Algorithm NSGA-II (Non-dominated Sorting Genetic Algorithm-II) [39] for *multi-objective* optimization (MOO) has been used previously in automated negotiation for QoS of web service applications [57]. What makes our approach novel in this regard is that we combine NSGA-II with an MCDM method called TOPSIS (Technique for Order Preference by Similarity to the Ideal Solution) [64] to choose the best among a set of ranked near Pareto-optimal outcomes during negotiation.

Fuzzy-based MOO

We further explore the idea of using the extended-NSGA-II of Bahri et al. [19] to deal with the MOO problem characterized by the necessity to simultaneously optimize two conflicting objectives. Consider, for instance, maximizing the user's utility and the opponent's utility in order to reach a 'win-win' solution in the presence of uncertain input data (i.e., reflecting the uncertainties in estimated user and opponent models). We hybridize this extended extended-NSGA-II with fuzzy TOPSIS [64] to choose the best among a set of ranked near Pareto-optimal outcomes during negotiation. We have seen this amalgamation of extended-NSGA-II and fuzzy-TOPSIS only in an application called supplier selection and multi-product allocation order problem [104]. Also, in [92], the prioritized fuzzy constraints were incorporated by Luo et al. into a buyer-seller negotiation setting, and the negotiation problem was considered as a fuzzy constraint-satisfaction problem. However, we deal with the fuzziness in the objective/utility functions of negotiating parties. So, to the best of our knowledge, we are the first to introduce and study fuzzy/non-fuzzy MOO with fuzzy/non-fuzzy MCDM in multi-issue bilateral negotiation to generate near-Pareto-optimal outcomes between two negotiating agents under their preference uncertainties.

Since the agent attempts to approximate the real preferences of both the negotiating parties, there is uncertainty involved. Hence, the MOO with the set of objective functions that may depend on uncertainty scenarios U_{sc} can be defined as follows [19]:

$$\max_{\omega\in\Omega,\xi\in U_{sc}} \quad (\widehat{U}_u(\omega,\xi), \widehat{U}_o(\omega,\xi)) \tag{2.8}$$

The cost of evaluating each uncertain objective function is represented by intervals such as a Triangular Fuzzy number (TFN). Formally, a TFN is represented with a triplet of values $A = [\underline{a}, \widehat{a}, \overline{a}]$, where $[\underline{a}, \overline{a}]$ is the interval of possible values called its support and $\widehat{a}$ denotes the most plausible (also known as its modal or kernel value). TFN can also be deduced from transformations of other different shapes by linguistic modifiers, compositions, projections and other operations. See [19] for more details. The triangular fuzzy-MOO problem of generating (near) Pareto-optimal bids in bilateral negotiation domain can be defined as a vector of objective functions ($\widehat{U}_u(\omega^\tau)$, $\widehat{U}_o(\omega^\tau)$), which are disrupted by the triangular form τ such that $\tau \in \mathbb{R}$ as shown in (2.9). In this objective space, the vector can be defined as a fuzzy cost function that represents the fitness of solutions or bids in terms of a triangular-valued objective vector such that $\widehat{U}_u(\omega^\tau) = [\underline{\widehat{U}_u(\omega)}, \widehat{\widehat{U}_u(\omega)}, \overline{\widehat{U}_u(\omega)}]$ and $\widehat{U}_o(\omega^\tau) = [\underline{\widehat{U}_o(\omega)}, \widehat{\widehat{U}_o(\omega)}, \overline{\widehat{U}_o(\omega)}]$.

$$\max_{\omega^\tau,\omega\in\Omega} \quad (\widehat{U}_u(\omega^\tau), \widehat{U}_o(\omega^\tau)) \tag{2.9}$$

2.2.7 Multiple-Criteria Decision-Making methods

Multi-Criteria Decision-Making (MCDM) methods have proven their effectiveness in addressing different complex decision-making problems where there is more than one conflicting criterion [93]. In this work, we use a MOO approach called NSGA-II [39]) to generate a list of Pareto-optimal solutions, which is followed by an MCDM method called TOPSIS [63] to let the agent decide one of them while considering two different objective functions (U_u and U_o) together by ranking the different alternatives (Pareto solutions) being evaluated during the negotiation. We choose TOPSIS among many other MCDM techniques available in the literature as it is simple, reliable, intuitive and easy to compute [143].

2.2.8 Gaps in the existing state-of-the-art

As evidenced from the above-mentioned existing work, automated negotiation has been an active area of research for over the past two decades. However, the extant work has a number of limitations, which we summarize below:

- Although learning approaches are proposed for automated negotiation, work in the domain of concurrent bilateral negotiations is missing.
- Existing DRL-based approaches used in automated negotiation strategies do not support continuous action spaces.
- No use of dynamic threshold utility is seen in the multi-issue negotiation literature under the BOA architecture.
- No "strategy templates" for acceptance and bidding strategies are explored in the extant literature to the best of our knowledge.
- The combined uncertainties of the approximated user and opponent modelling has never been considered together while generating Pareto-optimal bids.
- The amalgamation of MOO and MCDM methods has never been considered in the domain of negotiation for generating Pareto-optimal bids.
- The fuzzy approaches have not been explored to address the uncertainties in estimated utility models of both the negotiation parties while generating Pareto-optimal bids.

There are also other limitations of the existing work, such as, less consideration of preferential dependencies in multi-issue negotiation, minimal use of non-linear utility functions and negotiation protocol which is more complex than alternating offers protocol, and no exploration of learning approaches for the user preference elicitation process to reduce the user bother cost. However, they are beyond the scope of this thesis.

2.3 Negotiation Simulation Platforms

We perform experiments related to single-issue concurrent and multi-issue non-concurrent negotiations on two different simulation platforms called RECON [1] and GENIUS [88] respectively. We use RECON because it allows us to run concurrent bilateral negotiations. We also employ GENIUS as it gives access to many other strategies and a number of domains to perform experiments and compare results.

2.3.1 RECON

RECON stands for *Robust multi-agent Environment for simulating COncurrent Negotiations*. It supports the development of software agents interacting concurrently with other agents in a negotiation domain. It also supports declarative strategies, for applications where logic-based agents need to explain their negotiation decisions to a user. It consists of a set of infrastructure agents that can manage an electronic market and extract statistics from the negotiations that take place [1].

In general, RECON consists of the following three phases as shown in Figure 2.5:

- Phase 1 (*Configuration*) allows the user to define simulation parameters such as market density, market ratio, Zone of Agreement (ZoA), deadline, number of simulation runs, types of negotiating agents and their initial and reservation prices.

- Phase 2 (*Simulation*) conducts the actual negotiations between market agents (buyer agents and seller agents) based on the information from Phase 1 (*Configuration*). These negotiations are managed with the help of two infrastructure agents called market controller and market broker. The prime role of the market controller is to oversee the simulations. This involves initializing the simulations, creating the market agents and saving the negotiation logs at fixed time intervals. The market broker helps in notifying each agent about the entry of new agent in an e-market.

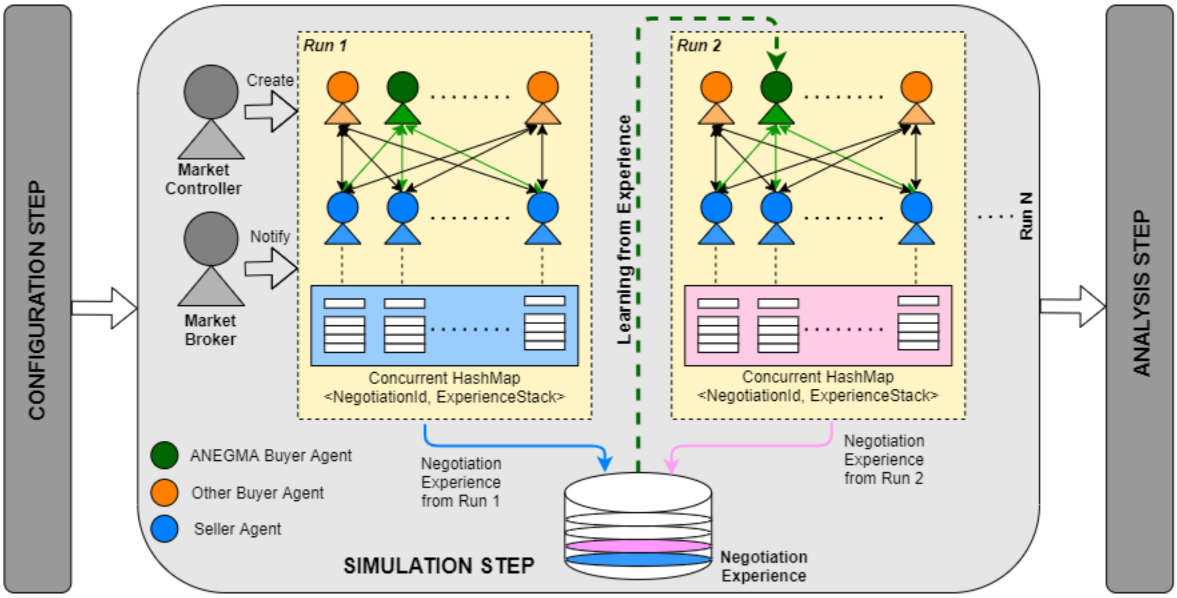

Figure 2.5: Modified Architecture of *RECON* simulation environment. Here, buyer agent stores the negotiation experience w.r.t. different sellers concurrently from each run in a global database (*Negotiation Experience*) to use the updated negotiation strategy (learned from experiences) in new simulation runs.

- Finally, Phase 3 (*Analysis*) analyses the negotiation logs to evaluate the performance of negotiations in terms of the various metrics defined by the user, such as average utility rate and average negotiation time.

We have substantially extended the RECON environment by adding a learning component to Phase 2 (*Simulation*), motivated by our proposed model (see Chapter 3). We will see later, in Chapter 3, that during each simulation run, our buyer agent maintains a concurrent hash map of negotiation IDs and a stack of past experiences while negotiating with different sellers concurrently, which is eventually added to a global memory *Negotiation Experience* at the end of each run. These past experiences are used by the buyer agent to learn and update the negotiation strategy and use it during new simulation runs. We have further extended RECON's market controller to handle the whole learning process during all simulations.

2.3.2 GENIUS

GENIUS stands for *General Environment for Negotiation with Intelligent multi-purpose Usage Simulation.* It helps facilitate both the design and evaluation of automated negotiators' strategies. It implements an open architecture that allows easy development and integration of existing negotiating agents and can be used to simulate individual negotiation sessions, as well as tournaments between negotiating agents in various negotiation scenarios. GENIUS also allows the specification of different negotiation domains and preference profiles by means of a graphical user interface [88]. GENIUS incorporates several mechanisms that aim to support the design of a general automated negotiator as shown in Figure 2.6. The first mechanism is an *analytical toolbox*, which provides a variety of tools to analyse the performance of agents, the outcome of the negotiation and its dynamics. The second mechanism is a *repository of domains and utility functions* which let the user define the preference profiles for their agents. Lastly, it also comprises *repositories of automated negotiators and negotiating protocols.* In addition, Genius enables the evaluation of different strategies used by automated agents that were designed using the tool. In Figure 2.6, *Simulation control* and *Logging* components will allow users to control and debug the simulations , as well as obtain the information [88]. This is an important contribution as it allows researchers to empirically and objectively compare their agents with others in different domains and settings [12]. GENIUS tool has also been supporting the ANAC[8] (Automated Negotiating Agents Competition) competition since 2010. ANAC is an international annual event which brings together researchers from negotiation community and provides unique benchmarks for evaluating practical negotiation strategies in multi-issue domains [70].

2.3.3 Other platforms

There are also some other negotiation frameworks/simulators in the literature such as: IAGO (*I*nteractive *A*rbitration *G*uide *O*nline (for Human-Agent negotiations) [97]),

[8] http://ii.tudelft.nl/nego/node/7

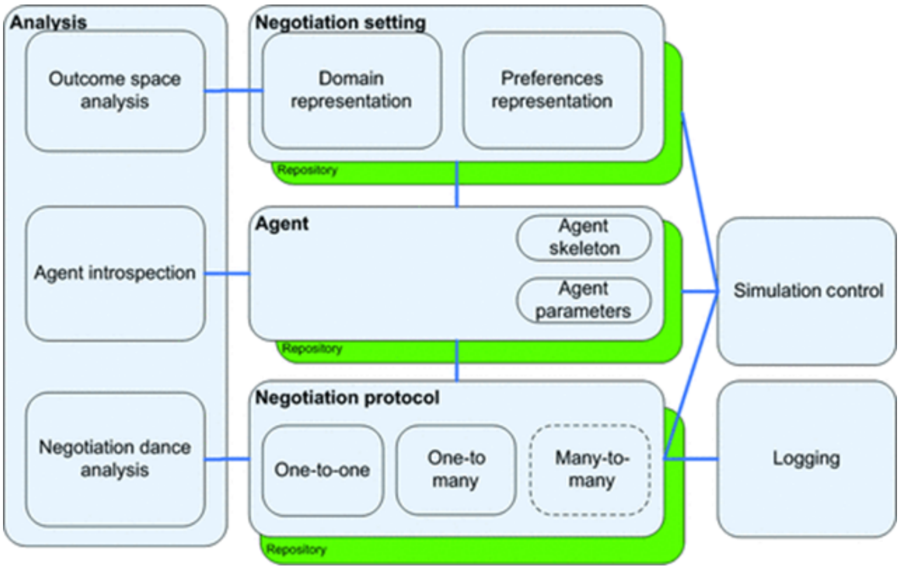

Figure 2.6: A High-level Architecture of GENIUS simulation platform [88]

BANDANA (*BA*sic e*N*vironment for *D*iplomacy playing *A*utomated *N*egotiating *A*gents) [69], Negowiki [94], Jupiter [50], MAN-REM (*M*ulti-*A*gent *N*egotiation and *R*isk management in *E*lectricity *M*arkets) [110], MASCEM (*M*ulti-*A*gent *S*imulator of *C*ompetitive *E*lectricity *M*arkets) [111], EMCAS (*E*lectricity *M*arket *C*omplex *A*daptive *S*ystem) [107], DESIRE [73], and Pocket Negotiator [71]. We do not consider them because they cannot support concurrent bilateral negotiations, unlike RECON, or they are not widely used and stable, and are domain-dependent, unlike GENIUS.

2.4 Summary

This chapter presented the background knowledge on the key subject areas underpinning this thesis. In particular, it explained the terminology used in the domain of automated negotiation, including the negotiation phases and the classification of negotiation. It also discussed the need of learning in negotiation. Moreover, it presented the existing work related to ours and identified gaps in the state-of-the-art. We observed the lack of specialized research on DRL-based negotiation strategies, as well as single and multi-objective optimization techniques for user preference modelling and generation of the Pareto-optimal bids, respectively. Finally, it gave a short overview of two widely-known negotiation simulation platforms used in this thesis for experimental evaluation. In subsequent chapters, we will develop our own agent negotiation models, which build upon some existing work discussed in this chapter.

Chapter 3

Single-Issue Bilateral Negotiation Model

In this chapter, we present *ANEGMA*, our first agent model that deals with concurrent bilateral negotiations in an e-marketplace like E-bay. We start by discussing the negotiation environment of *ANEGMA* and its associated settings in Section 3.1. Then, in Section 3.2, we present the details of the model based on DRL to address indivisible single-issue bilateral negotiation. In Section 3.3, we describe how we generate synthetic supervision data for an *ANEGMA* agent to learn from 'teacher strategies', we identify performance measures, and describe the employed SL and DRL models. We, then, experimentally evaluate our proposed work by analysing the results we obtain by playing our agent against the state-of-the-art in Section 3.4, which is further followed by the summary of our conclusions in Section 3.5.

3.1 Negotiation Environment

In our work, we consider e-marketplaces like E-bay where the competition is visible, i.e., a buyer agent can observe the number of competitors that are dealing with the same resource from the same seller. In particular, we assume that the environment E consists of a single e-market m with P agents, with a non-empty set of buyer agents B_m and a non-empty set of seller agents S_m – these sets need not be mutually exclusive. For a buyer $b \in B_m$ and resource r, we denote with $S^t_{b,r} \subseteq S_m$ the set of seller agents from the market m which, at time point t, negotiate with b for a resource r over a range of issues I. The buyer agent b uses $|S^t_{b,r}|$ negotiation threads, in order to negotiate concurrently with each seller in $S^t_{b,r}$. We further assume that no agent can be both buyer and seller for the same resource at the same time, that is, $\forall b, r, t.\ s \in S^t_{b,r} \implies S^t_{s,r} = \emptyset$. The set $C^t_{b,r} = \{b' \neq b \in B_m \mid S^t_{b',r} \cap S^t_{b,r} \neq \emptyset\}$ includes the competitors of b, i.e., those agents negotiating with the same sellers and for the same resource r as those of b.

We adopt the negotiation protocol of [2] because it reflects many realistic negotiation scenarios in open e-markets, as follows. The buyer and seller have their own private deadlines, they can negotiate in concurrent bilateral negotiations, and they may be aware of competition (e.g., as in E-Bay[1]). Also, the protocol allows for actions where a buyer can show interest for a product (e.g., as in Shpock[2]), to the extent that it can reserve it for a period of time, with a penalty (deposit) if the reservation is cancelled. In general, a negotiation protocol describes the set of rules that each buyer b and seller s should follow during a negotiation thread, including the valid moves agents can take at any state of the negotiation. The protocol is known to all agents in advance. The protocol, illustrated in Figure 3.1, assumes an open e-market environment, i.e., where agents can enter or leave the negotiation at their own will. We assume each negotiation focuses on a single resource characterized uniquely by a class and a fixed, non-negotiable, set of properties. The class 'laptop' with properties 'Lenovo/16 GB RAM/500 GB Hard disk' is an example of a resource r which can be used during negotiation between two agents. For such a resource r, we negotiate over a single issue, namely, price. We further assume that negotiation is represented internally for a buyer agent as a dialogue with a unique identifier so that the agent can distinguish between different negotiations for the same resource originating from different sellers. It is beyond the scope of this work

[1] `https://ebay.com`

[2] `https://www.shpock.com`

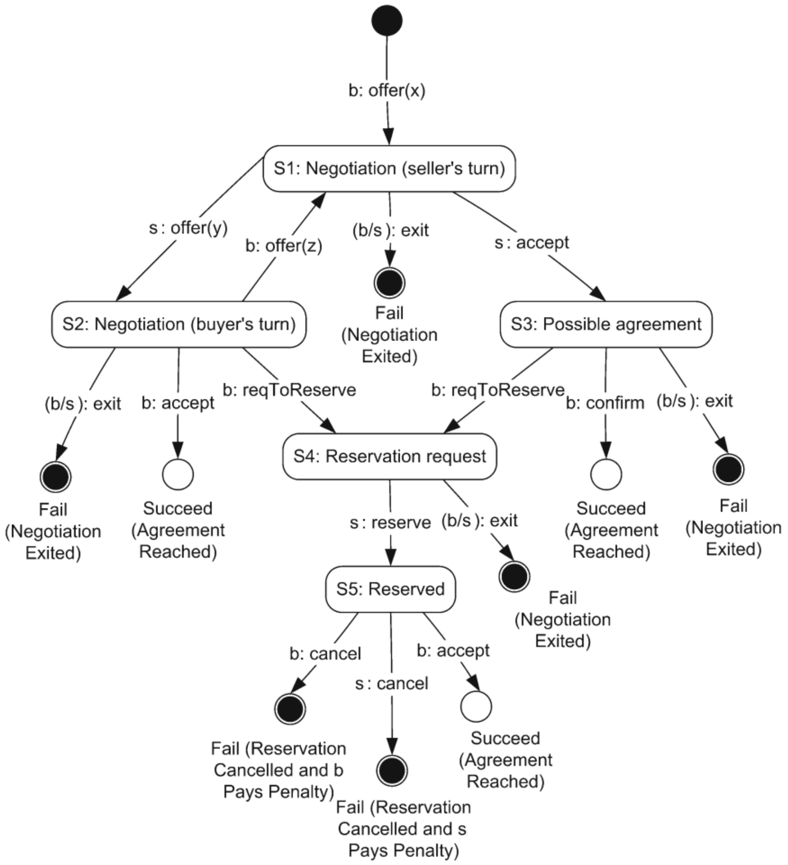
b: offer(x)
S1: Negotiation (seller's turn)
s : offer(y)
b: offer(z)
(b/s): exit
s : accept
S2: Negotiation (buyer's turn)
Fail (Negotiation Exited)
S3: Possible agreement
(b/s): exit
b: accept
b: reqToReserve
b: reqToReserve
b: confirm
(b/s): exit
S4: Reservation request
Fail (Negotiation Exited)
Succeed (Agreement Reached)
s : reserve
(b/s): exit
Succeed (Agreement Reached)
Fail (Negotiation Exited)
S5: Reserved
Fail (Negotiation Exited)
b: cancel
b: accept
s : cancel
Fail (Reservation Cancelled and b Pays Penalty)
Succeed (Agreement Reached)
Fail (Reservation Cancelled and s Pays Penalty)

Figure 3.1: Negotiation Protocol [2]

to deal with multiple resources.

Furthermore, we assume that a buyer agent b always starts the negotiation by making an offer. With t_{start} we denote the start time of the negotiation, and with t_b the maximum duration of any negotiation, which for simplicity, is the same for all the agents during all the negotiation sessions irrespective of which resource these agents are negotiating for. The deadline for b is, thus, $t_{end} = t_{start} + t_b$. Information about the deadline t_b, Initial Price IP_b (we assume that $IP_b > 0$) and Reservation Price RP_b is private to each $b \in B_m$. Each seller s also has its own private Initial Price IP_s, and Reservation Price RP_s. In other words, each agent has a private aspiration zone, which is a maximum or minimum range that must be respected in order to reach a deal. The intersection between the agents' aspiration zones is known as a bargaining zone or Zone of Agreement (ZoA). This is shown in Figure 3.2, and it is denoted by Z. Here, $Z = [IP_b, RP_b] \cap [IP_s, RP_s]$ which is an overlapping re-

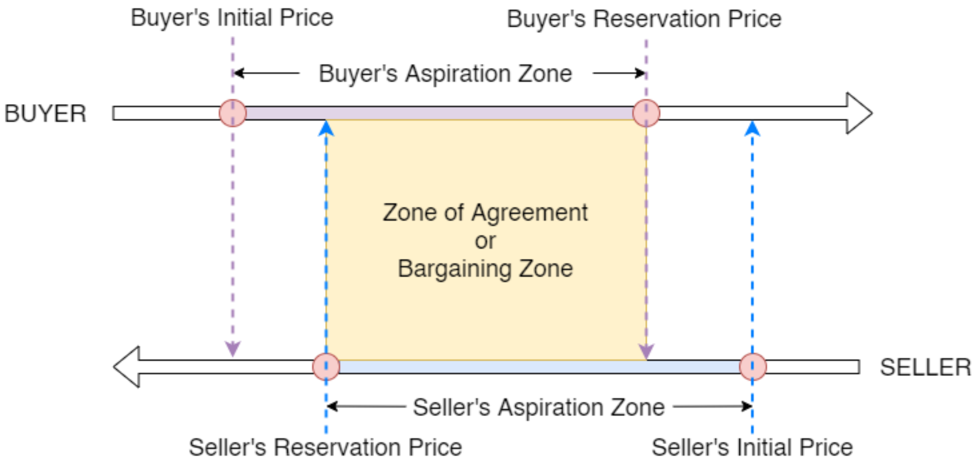

Figure 3.2: Zone of Agreement

gion between buyer's and seller's reservation prices. Both parties reach a successful agreement if $Z \neq \phi$. The protocol is turn-based and allows agents to take actions from a pool *Actions* at each negotiation state (from S1 to S5, see Figure 3.1):

$$Actions = \{offer(x), reqToReserve, reserve, cancel, confirm, accept, exit\}, \qquad (3.1)$$

where

- *offer(x)*: The offer made by b or s, where x is the price.

- *accept*: On performing this action, b or s agrees to the last offer made by their counterpart. When performed by the buyer, an *accept* leads to successful completion of the negotiation (see states S2 and S5). When performed by the seller, the buyer either acknowledges it with a *confirm* action or can buy more time with a *reqToReserve* action.

- *reqToReserve*: After s makes a counter-offer (state S2) or accepts b's offer (state S3), b can perform this action to request s to reserve the resource with the latest offer. By not committing immediately to accepting the offer, b can wait for a better offer from another seller (and negotiation thread) $s' \in S^t_{b,r} \setminus \{s\}$.

- *reserve*: It is used by s to acknowledge and agree to a *reqToReserve* action from b.

- *cancel*: It allows both b and s to cancel their reserved offers. The cancelling agent pays a penalty to the other negotiating agent to avoid unnecessary cancellations and bias. Cancelling leads to no agreement.

- *confirm*: With this action, buyer b acknowledges that the seller has accepted b's offer, and the negotiation terminates with an agreement. When dealing concurrently with different sellers for the same resource, b is allowed to send a *confirm* action only to one seller to reach an agreement.

- *exit*: It allows both b and s to withdraw from the negotiation at any time (without notifying the opponent) implying that negotiation has failed.

An outcome is either *Fail* if b or s performs an *exit* or *cancel*; or it is *Succeed* if b accepts or confirms the current offer.

At any time point t, during negotiation, the following information about the state of the environment can be identified:

- $s_b^t\{^{i}_{jk}\} \in S^i_{jk}$ is a set of seller agents from the market i negotiating with a buyer b at any time t for a resource $j \in R$ over an issue $k \in I$.

- The buyer agent b has $|s_b^t\{^{i}_{jk}\}|$ negotiation threads, one for each seller $\in s_b^t\{^{i}_{jk}\}$, thus negotiating concurrently with $|s_b^t\{^{i}_{jk}\}|$ sellers at time t. This relationship between buyers and sellers can also be represented using a bipartite graph, as shown in Figure 3.3. In Figure 3.3, the number of concurrent negotiation threads for a buyer b_1 refers to the degree of buyer node b_1, which is equal to 3.

- C_b^t = set of competitor agents for buyer $b \in B$ at any time t, where $C_b^t \subseteq B \setminus \{b\}$ negotiating with same seller for same resource over same issue. In Figure 3.3, b_2 and b_4 are competitor agents for b_1 when dealing with s_1 and s_4 respectively.

- $[IP_b, RP_b]$ represents the Initial Price and Reservation Price of buyer agent $b \in B$. This information is private to each agent b.

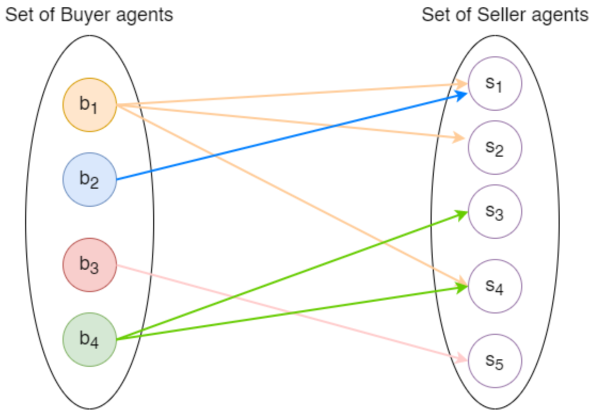

Figure 3.3: Bipartite Graph showing relation between buyers and sellers

Capping all above, for our research study, we have considered an environment consisting of a single e-market with visible competition for bilateral negotiation where each buyer is negotiating with different sellers for only one resource which is associated with one issue, i.e., *single-issue single-resource negotiation*.

3.2 The *ANEGMA* Model

When a negotiating agent enters the e-market, it will usually be surrounded by multiple opponents offering its preferred resource, which reflects what happens in real-life situations. When the agent starts to negotiate with multiple opponents at the same time, the three main challenges are: firstly, how to maximize the agent's utility by selecting the negotiation with the opponent that will offer the best agreement within a certain time limit; secondly, how to manage the ongoing concurrent negotiation threads; and lastly, how to scrutinize the effect of progress in negotiation with one opponent on the progress of negotiations with other opponents. In this context, we introduce our proposed negotiation model called *ANEGMA* (*A*daptive *NEG*otiation model for e-*MA*rkets) and explain its components.

3.2.1 *ANEGMA* Components

Our proposed agent negotiation model supports learning during concurrent bilateral negotiations with unknown opponents (or opponent strategies) in dynamic and

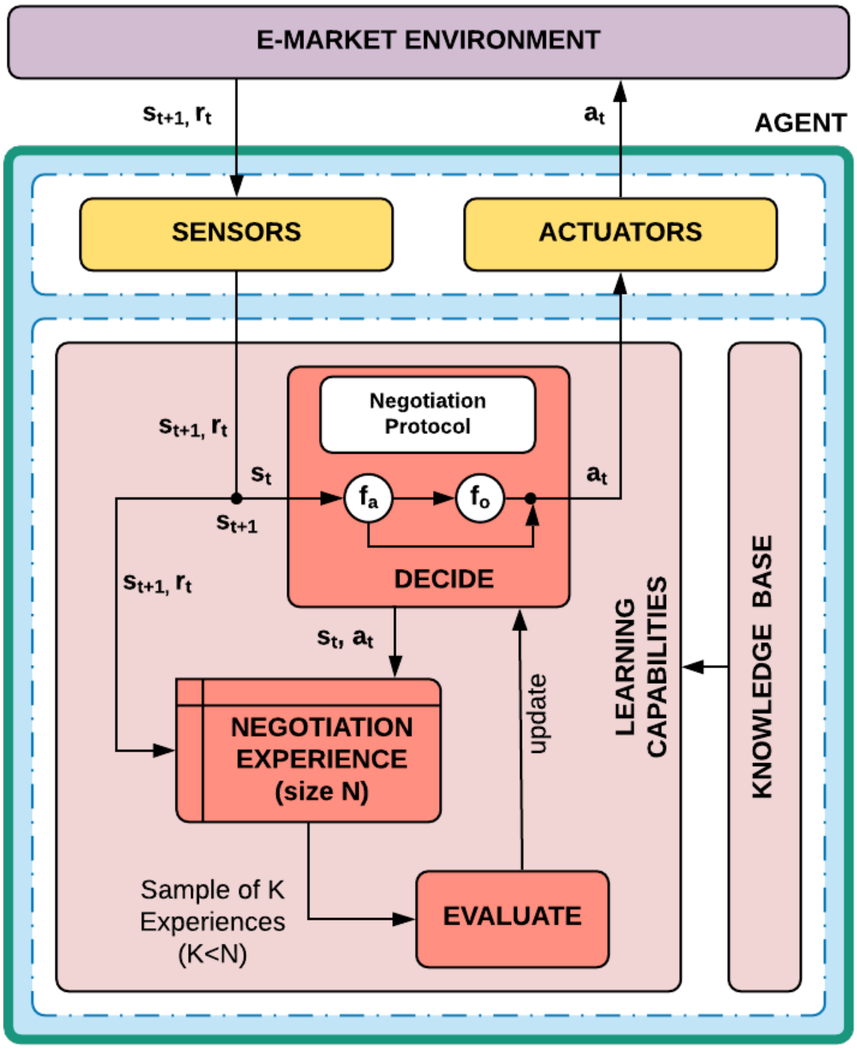

Figure 3.4: The Architecture of *ANEGMA*

complex e-marketplaces. This in unlike game-theoretic approaches where each party is aware of other's strategies [157]. In this model, we use a centralized approach in which the coordination is done internally to the agent via multi-threading synchronization. This approach minimizes the agent communication overhead and thus, improves the run-time performance. The different components of the proposed model are shown in Figure 3.4 and explained below.

Physical Capabilities

The *sensors* of the agent enable it to access an e-marketplace. They allow a buyer b to perceive the current (external) state of the environment s_t and represent that state locally in the form of internal attributes, as shown in Table 3.1. Some of

Table 3.1: Agent's State Attributes

Attribute	Description
NS_r	Number of sellers that b is concurrently dealing for resource r at time t ($\lvert S^t_{b,r}\rvert$).
NC_r	Number of buyer agents competing with b for resource r at time t ($\lvert C^t_{b,r}\rvert$).
S_{neg}	Current state of the negotiation protocol (S1–S5, see Figure 3.1).
X_{best}	Best offer made by either b or s in S_{neg}.
T_{left}	Time left for b to reach t_{end} after the last action of s.
IP_b	Minimum price which b can offer at the start of the negotiation.
RP_b	Maximum price which b can offer to s.

these attributes (NS_r, NC_r) are perceived by the agent using its sensors, some of them (IP_b, RP_b, t_{end}) are stored locally in its knowledge base and some of them (S_{neg}, X_{best}, T_{left}) are obtained while interacting with other seller agents during a negotiation. At time t, the internal agent representation of the environment is s_t, which is used by the agent to decide what action a_t to execute using its *actuators*. Action execution then changes the state of the environment to s_{t+1}.

Learning Capabilities

The foundation of our model is a component providing learning capabilities similar to those in the Actor-Critic architecture of [86]. It consists of three sub-components: *Negotiation Experience*, *Decide* and *Evaluate*.

Negotiation Experience stores historical information about previous *negotiation experiences* N which involve the interactions of an agent with other agents in the market. Experience elements are of the form $\langle s_t, a_t, r_t, s_{t+1}\rangle$, where s_t is the internal representation of the e-market environment state perceived by the agent at time t, a_t is an action performed by b at s_t, r_t is a scalar reward or feedback received from the environment and s_{t+1} is the new e-market state after executing a_t.

The negotiation strategy is enacted by the *decide* component. At any given state s_t, the strategy determines the optimal action for b, choosing among the available set of actions *Actions*, see (3.1). In particular, the strategy builds on two functions f_a and f_o. Function f_a takes state s_t as an input and returns a discrete action among *offer(x), accept, confirm, reqToReserve* and *exit*, see (3.2). When f_a decides to perform an *offer(x)* action, f_o is used to compute, given an input state s_t, the value of x, see (3.3). The functions f_a and f_o belong to the Machine Learning decision box of the sequence diagram presented in Figure 3.5.

$$f_a(s_t) = a_t, \quad \text{where } a_t \in \mathit{Actions} \tag{3.2}$$

$$f_o(s_t) = x, \quad \text{where } x \in [IP_b, RP_b] \tag{3.3}$$

Evaluate refers to a critic which helps b learn and evolve the strategy for unknown and dynamic environments. It is a function of K (where $K < N$) randomly selected past negotiation experiences. The learning process of b is *retrospective* since it depends on the feedback (or scalar rewards) obtained from the e-market environment by performing action (either discrete or continuous) a_t at state s_t. Our design of reward functions accelerates agent learning by allowing b to receive rewards after every action it performs in the environment instead of receiving only at the end of the negotiation. The reward at time t, r_t is given by:

$$r_t = \begin{cases} U_b(x,t), & \text{if Succeed} \\ -1, & \text{if Fail} \\ r'_t & \text{if } a_t = \text{offer}(x) \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

$$r'_t = \begin{cases} U_b(x,t), & \text{if } x \le \min(O_t) \\ -1, & \text{if } x > \min(O_t) \end{cases} \tag{3.5}$$

$$U_b(x,t) = \left(\frac{RP_b - x}{RP_b - IP_b}\right) \cdot \left(\frac{t}{t_{end}}\right)^{d_t} \tag{3.6}$$

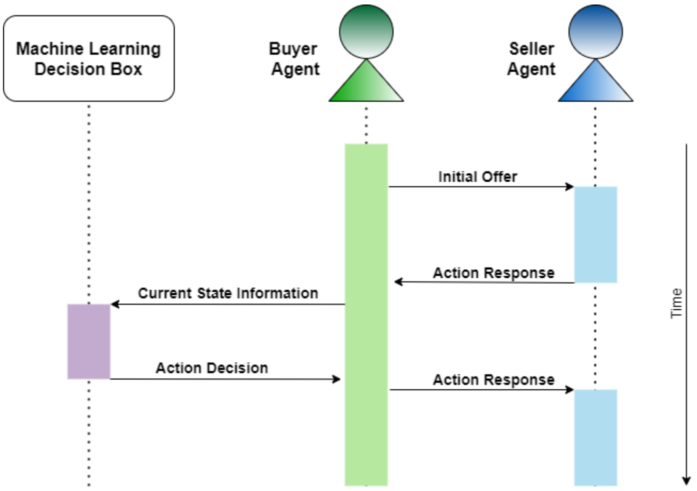

Figure 3.5: Sequence diagram of *ANEGMA*

The reward values r_t and r'_t computed in (3.4) and (3.5) evaluate the discrete action decided by f_a and continuous action decided by f_o at time t respectively. Function $U_b(x, t)$, see (3.6), refers to the utility of offer $x = f_o(s_t)$ at time t and is calculated using Initial Price (IP_b), Reservation Price (RP_b), offer x, and a temporal discount factor $d_t \in [0, 1]$ to penalize delays in negotiation, which was set to 0.6 in our experiments. Higher d_t value implies higher penalty due to delay. In other words, the discount factor reduces the utility of deals with the progression of time. The reward r'_t in (3.5) helps b learn that it should not offer more than what active sellers have already offered: O_t is a list of preferred offers received from sellers $s \in S^t_{b,r}$ at time t, which b maintains during the negotiation. To sum up, our reward function is designed to encourage (i.e., returns a positive reward value) our agent to conclude a successful negotiation timely and discourage (i.e., returns a negative reward value) no deal or when our buyer agent offers more than any of the offers proposed by active sellers. Otherwise, it is neutral to all other actions (i.e., returns 0 reward).

3.3 Setting Up *ANEGMA* for Experiments

In our approach, we first use SL to pre-train the *ANEGMA* agent using supervision examples collected from existing negotiation strategies. Such pre-trained strategy is then evolved via RL using experience and rewards collected while interacting with other agents in the negotiation environment. This combination of SL and RL approaches enhances the process of learning an optimal strategy. This is because applying RL alone from scratch would require a large amount of experience before reaching a reasonable strategy, which might hinder the online performance of our agent. On the other hand, starting from a pre-trained policy ensures quicker convergence (as demonstrated empirically in Section 3.4). In this section, we describe the methods for collection of supervision examples and the relevant learning techniques.

3.3.1 Data set collection

In order to collect the data set for pre-training the *ANEGMA* agent via SL, we have used the *RECON* simulation environment [1] as discussed in subsection 2.3.1. A key advantage of this solution is that we can generate arbitrarily large sets of synthetic negotiation data, and for different choices of buyer and seller strategies. While, in principle, real-world market data could be used for this purpose as well, to the best of our knowledge, no publicly available datasets exist that fit our settings. In particular, in our experiments, we generate supervision data using the buyer strategies of [2] and [148] (see Section 3.4).

RECON supports concurrent negotiations between buyers and seller agents and is built on the top of GOLEMlite [101],which is a Java library for managing e-markets and extract relevant negotiation statistics.

3.3.2 Strategy representation

We represent both strategies f_a and f_o (see Equations 3.2 and 3.3) using ANNs [53], as these are powerful function approximators and benefit from extremely effective learning algorithms. From a machine learning perspective, approximating f_a amounts to solving a *classification* problem because of f_a's discrete output domain, see (3.2). On the other hand, approximating f_o corresponds to solving a *regression* problem because of f_o's continuous output, see (3.2).

ANN In particular, we use feed-forward neural networks, i.e., functions organized into several layers, where each layer comprises a number of neurons that process information from the previous layer. Formally, let l be the total number of layers in the network, which includes $l-1$ hidden layers and one output layer. Let n_i be the number of neurons in layer i ($i = 1, \ldots, l$), where n_0 be the number of neurons in the input layer (i.e., the input dimensionality). For input $x \in \mathbb{R}^{n_0}$, the function computed by a feed-forward neural network F is

$$F(x) = f^{(l)}\left(f^{(l-1)}\left(\ldots\left(f^{(1)}(x)\right)\ldots\right)\right) \tag{3.7}$$

where $f^{(i)} : \mathbb{R}^{n_{i-1}} \rightarrow \mathbb{R}^{n_i}$ is the function computed by the i-th layer, which is given by

$$f^{(i)}(p_{i-1}) = g^{(i)}(W^{(i,i-1)} \cdot p^{(i-1)} + b^{(i)}), i = 1, \ldots, l \tag{3.8}$$

where $p_{i-1} \in \mathbb{R}^{n_{i-1}}$ is the output vector of layer $i-1$, $W^{(i,i-1)} \in \mathbb{R}^{n_i \times n_{i-1}}$ is the weight matrix connecting p_{i-1} to the neurons of layer i, $b^{(i)} \in \mathbb{R}^{n_i}$ is the bias vector of layer i, and $g^{(i)}$ is the activation function of the neurons of layer i. For our experiments, we used a soft max activation function for classification and a linear activation function for regression at the final output layer of our models f_a (i.e., classification (3.4)) and f_o (i.e., regression (3.5)) respectively.

Learning an ANN corresponds to finding values of its weights and biases that minimize a given loss function. The learnable parameters are typically updated via some form of gradient descent, where the gradient of the loss function w.r.t. the parameters is computed via back-propagation [53].

In supervised learning, the loss function captures the deviation between the supervision data and the corresponding model's predictions. We used cross-entropy and mean square error to approximate f_a and f_o respectively.

For each data sample $x \in \mathbb{R}^{n_0}$, the network's prediction is compared to the actual known target value of that data sample (discrete or continuous value). The function parameters (weights and biases) are also learned and modified during training so to minimize the loss. These modifications are performed in the backward direction from the output layer through each hidden layer down to the first hidden layer.

To reduce over-fitting and generalization error, during the training of the ANN we applied regularization techniques, drop-out in particular.

DRL When being in a state s_t, DDPG (as discussed in Chapter 2) uses a so-called *actor* network μ to select an action a_t, and a so-called *critic* network Q to predict the value Q_t at state s_t of the action selected by the actor:

$$a_t = \mu(s_t \mid \theta^{\mu}) \tag{3.9}$$

$$Q_t(s_t, a_t \mid \theta^{Q}) = Q(s_t, \mu(s_t \mid \theta^{\mu}) \mid \theta^{Q}) \tag{3.10}$$

In (3.9) and (3.10), θ^{μ} and θ^{Q} are, respectively, the learnable parameters of the actor and critic neural networks. The parameters of the actor network are updated by the Deterministic Policy Gradient method [129]. The objective of the actor policy function is to maximize the expected return J calculated by the critic function:

$$J = \mathbb{E}[Q(s, a|\theta^{Q})|_{s=s_t, a=\mu(s_t)}]. \tag{3.11}$$

To this purpose, the parameters of μ are updated (via gradient ascent) using the gradient of J w.r.t. the actor policy parameters. In particular, the expectation in (3.11) is approximated using the average of K randomly selected past experiences (or mini-batches) (s_i, a_i, r_i, s_{i+1}).

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{K} \sum_{i=1}^{K} \left[\nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})|_{s=s_i} \right] \tag{3.12}$$

The critic network Q should predict the expected return obtained by performing action a_t at state s_t and thereafter follow the policy entailed by the actor μ. For this purpose, [85] shows that Q can be derived, using K random mini-batches, by minimizing the following loss function:

$$L = \frac{1}{K}\sum_{i=1}^{K}(y_i - Q(s_i, a_i|\theta^Q))^2, \text{ where} \tag{3.13}$$

$$y_i = r_i + \gamma Q(s_{i+1}, \mu(s_{i+1}|\theta^\mu)|\theta^Q), \tag{3.14}$$

and $\gamma \in (0, 1)$ is a discount factor. Since the target Q-value y_i used to update Q depends on Q itself, which might cause divergence, DDPG employs two additional neural networks called actor target network μ' and critic target network Q' in place of μ and Q in (3.14). These are copies of μ and Q which are updated in a soft manner, i.e., by slowing tracking μ and Q rather than exactly copying them, which the effect of regularizing learning and increasing stability. See [85] for further details.

3.4 Experimental Setup and Results

In this section, we experimentally evaluate our *ANEGMA* approach in negotiations against unknown opponents during concurrent bilateral negotiations in different e-market settings.

3.4.1 Experimental Settings

We consider the following buyer strategies:

- *CONAN [2]:* A heuristic strategy which uses a weighted combination of agent's internal state attributes as well as environmental parameters (in which agent situates) to calculate the concession rate. This strategy lets the agent negotiate with multiple sellers concurrently without any additional coordinator.

- *Williams [148]:* A strategy performing Gaussian process regression for predicting the seller agent's future utility while bidding a counter-offer. This strategy is originally used to negotiate with multiple opponents for the same item over multiple issues with the help of a coordinator, which is responsible for finding the best of all deals with different opponents based on time and utility.

- *SL-C*[3].: An ANN-based strategy obtained using supervised learning from CONAN data.
- *SL-W*[3]*:* An ANN-based strategy obtained using supervised learning from Williams' data.
- *DRL:* A DRL strategy initialized with an ANN with random parameters.
- *ANEGMA-C:* Our ANEGMA strategy obtained via DRL and initialized with the ANN pre-trained with CONAN data (SL-C).
- *ANEGMA-W:* Our ANEGMA strategy obtained via DRL and initialized with the ANN pre-trained with Williams data (SL-W).

For carrying out the experiments, we have used the RECON simulation environment [1] and extended it to support online agent learning, as shown in Figure 2.5. Since the Williams' strategy does not support 'reserve', 'reqToReserve' and 'cancel', we omit these actions from our experimental analysis for fair comparison.

Performance Evaluation Measures. To successfully evaluate the performance of *ANEGMA* (ANEGMA-C and ANEGMA-W) and compare it with other negotiation approaches, we selected the following widely adopted metrics [148, 42, 106, 2]: *Average utility rate* (U_{avg}), *Average negotiation time* (T_{avg}) and *Percentage of successful negotiations* $(S_{\%})$, which are described in Table 3.2.

Seller Strategies. We consider two widely-known and standard groups of fixed seller strategies developed by Faratin [42] as discussed briefly in Table 3.3: Time-Dependent and Behaviour-Dependent, each consisting of three different types of seller strategies. In time-dependent strategies (*Linear*, *Conceder* and *Boulware*),

[3] SL-X is identical to pre-training phase of ANEGMA-X, where X $\in \{C, W\}$

Table 3.2: Performance Evaluation Metrics

Metric	Definition	Ideal Value
U_{avg}	Total negotiation utility averaged over the successful negotiations.	High (1.0)
T_{avg}	Total time taken by the buyer agent (in milliseconds) averaged over all successful negotiations to reach the agreement.	Low ($\approx$ 1000ms)
$S_{\%}$	Proportion of successful negotiations	High (100%)

Table 3.3: Different Faratin's Seller Strategies

Time-Dependent	
Linear	The agent concedes the same amount during whole negotiation.
Conceder	The agent concedes a lot in the early phase of the negotiation.
Boulware	The agent keeps his initial offer almost until the deadline and concedes considerably only at the end.
Behaviour-Dependent	
Relative tit-for-tat	The agent reproduces, in percentage terms, the behaviour that its opponent performed in the previous rounds.
Random Absolute tit-for-tat	The agent behaves same as that of relative tit-for-tat, except that the behaviour is imitated in absolute terms.
Averaged tit-for-tat	The agent uses the average of the percentage change in a window (or slice) of the opponent's history.

the seller considers the remaining negotiation time for calculating the counter-offer value and the acceptance value for the offer received from the buyer. On the other hand, in behaviour-dependent strategies (*Relative tit-for-tat*, *Random Absolute tit-for-tat* and *Averaged tit-for-tat*), the seller imitates the observed behaviour of the buyers in order to compute the counter-offer. During experimentation, the same private deadlines were used for both sellers and buyer. Other parameters such as IP_s and RP_s are determined by the *ZoA* parameter, as shown in Table 3.4.

Competitor Strategies. All competitor strategies are chosen randomly between Simple Buyer (which generates offers randomly) and Nice Tit for Tat (which reproduces the opponent's behaviours of the previous negotiation rounds by reciprocating the opponent's concessions).

Simulation Parameters. We assume that the buyer negotiates with multiple sellers concurrently to buy a second-hand laptop ($r = laptop$) based only on a single issue *Price* ($I = \{Price\}$). We stress that the single-issue assumption is not unrealistic for e-markets like e-Bay, where sellers advertise a product with a fixed set of issues (e.g., Lenovo, 16 GB RAM, 250 GB HDD, i7 processor) and the only issue being negotiated is price. The simulated market allows the agents to enter and leave the market at their own will. The maximum number of agents allowed in the market, the demand/supply ratio, the buyer's deadline and the *ZoA*s are simulation-dependent.

As in [2], three qualitative values are considered for each parameter during simulations, e.g., High, Average and Low for *MD* or t_{end}. Parameters are reported in Table 3.4. The user can select one of such qualitative values for each parameter. Each qualitative value corresponds to a set of three quantitative values, of which only one is chosen at random for each simulation (e.g., setting *High* for parameter *MD* corresponds to choosing at random among 30, 40, and 50). The only exception is parameter *ZoA*, which maps to a range of uniformly distributed quantitative values for the seller's initial price IP_s and reservation price RP_s (e.g., selecting *Average* for *ZoA* leads to a value of IP_s uniformly sampled in the interval $[580, 630]$). Therefore, the total number of simulation settings is 81, as we consider 3 possible settings for each of *MD*, *MR*, t_{end}, and *ZoA* (see Table 3.4).

Neural Network Architecture. We represent the supervised learning policy as a neural network with 2 fully-connected hidden layers of 64 units and one output layer. The hidden layers use ReLU (Rectified Linear Unit) activation function whereas the output layer uses softmax and linear activation functions for classification and regression respectively. For DDPG, we represent deep neural networks with the same above mentioned neural networks.

3.4.2 Experimental Hypotheses

With our experiments, we aim to demonstrate the following hypotheses:

Hypothesis A: The *Market Density (MD)*, the *Market Ratio* or *Demand/Supply Ratio* (*MR*), the *Zone of Agreement* (*ZoA*) and the *Buyer's Deadline* (t_{end}) have a considerable effect on the success of negotiations. Here,

- *MD* is the total agents in the e-market at any given time dealing with the same resource as that of our buyer.
- *MR* is the ratio of the total number of buyers over the sellers in the e-market.
- *ZoA* refers to the intersection between the price ranges of buyers and sellers for them to agree.

In practice, buyers have no control over these parameters except the deadline (t_{end}), which can be decided by the user, or constrained by a higher-level goal the buyer is trying to achieve. While this hypothesis is not directly concerned with the performance of *ANEGMA*, it establishes that, for an adequate performance evaluation, it is necessary to fix a particular choice of these parameters. Otherwise, the performance variability will be too high to make any useful assessment.

Hypothesis B: The *ANEGMA* buyer outperforms the SL-only, DRL-only, CONAN, and Williams' negotiation strategies in terms of U_{avg}, T_{avg} and $S_{\%}$ in a range of e-market settings.

Hypothesis C: An *ANEGMA* buyer, if trained against a specific seller strategy, still performs well against other unknown seller strategies. This shows that the *ANEGMA* agent behaviour is *adaptive* in the sense that the agent transfers knowledge from previous experience to unknown e-market settings.

Table 3.4: Simulation Parameter Values

Parameter	**Values range**		
	100% ZoA (High)	60% ZoA (Average)	10% ZoA (Low)
IP_b	$[300 - 350]$	$[300 - 350]$	$[300 - 350]$
RP_b	$[500 - 550]$	$[500 - 550]$	$[500 - 550]$
IP_s	$[500 - 550]$	$[580 - 630]$	$[680 - 730]$
RP_s	$[300 - 350]$	$[380 - 430]$	$[480 - 530]$
MD	$\{30, 40, 50\}$	$\{18, 23, 28\}$	$\{8, 10, 12\}$
MR	{10:1, 1:1, 1:10}	{5:1, 1:1, 1:5}	{2:1, 1:1, 1:2}
t_{end}	[151s –210s]	[91s –150s]	[30s –90s]

3.4.3 Empirical Evaluation

We evaluate and discuss the three research hypotheses introduced at the beginning of the section.

Hypothesis A (*MD*, *MR*, *ZoA* and t_{end} Have Significant Impact on Negotiations)

We experimented with 81 different e-market settings over 500 simulations using the CONAN buyer strategy. Both time-dependent and behaviour-dependent seller strategies were considered for each setting. These experiments suggest that *MD* and *ZoA* have a considerable effect on $S_{\%}$ (Figure 3.6 - Figure 3.9). In Figure 3.6, we observe that the agents reach more negotiation agreements when *MD* is low. This suggests that in small markets offering the required resource, the number of successful deals is maximized, which in turn implies that being in a large market isn't always better. Also, there is not much difference in the agreement rate for 60% and 100% *ZoA* when *MD* is low. The small number of successful negotiations for 10% *ZoA* is not unexpected, since only a minority of agents is willing to concede more in such a small *ZoA*. On the other hand, *MR* and t_{end} have, according to our experiments, a comparably minor impact on the negotiation success (see Figures 3.8 and 3.9). Also, only some effect of *MR* on $S_{\%}$ is observed under low *MD* against behaviour-dependent strategies, as shown in Figure 3.7. Moreover, we performed significance tests (i.e., Z-tests for independent proportions) for all the relevant pair-

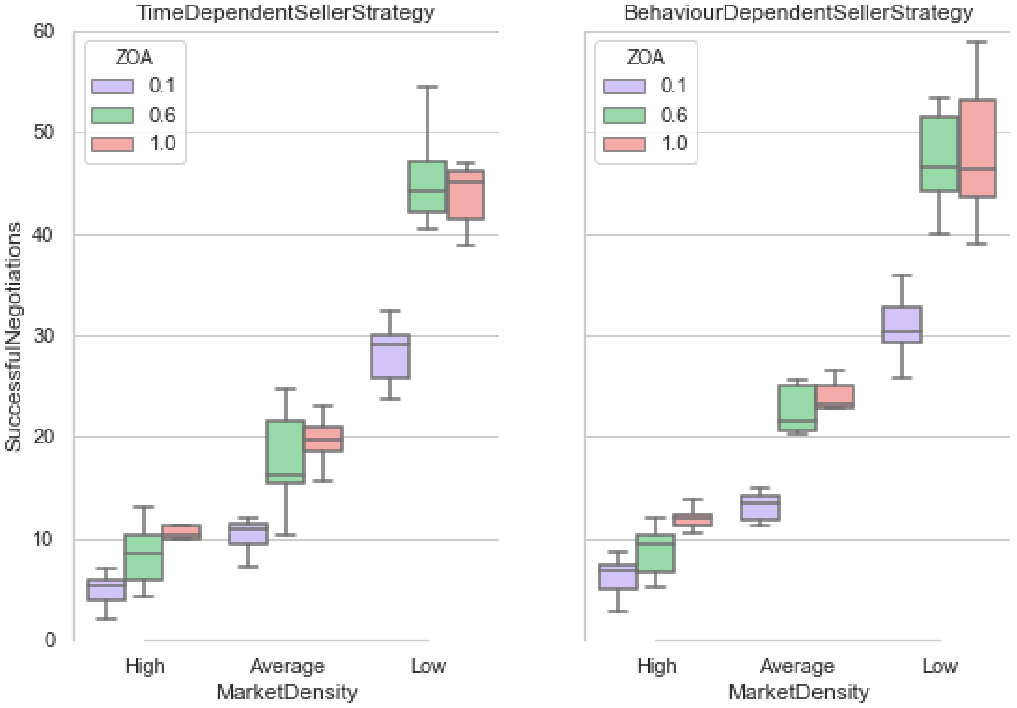

Figure 3.6: Effect of Market Density and ZoA on Percentage of Successful Negotiations

wise comparisons. All the differences in the proportions of successful runs were found significant at $p < 2.12E-13$.[4] Hence, these results support our hypothesis.

Hypothesis B (ANEGMA outperforms SL, CONAN, and Williams')

We performed simulations for our *ANEGMA* agent in low MD, 60% and 100% ZoA, high MR, and a long t_{end} because these settings yielded the best performance in terms of $S_{\%}$ in our experiments for Hypothesis A. To test how our strategy learns against two different categories of fixed seller strategies (i.e., Time-dependent and Behaviour-dependent) as well as to limit the experiments, we randomly choose *Conceder Time Dependent* and *Relative Tit for Tat Behaviour Dependent* seller strategies in the above simulation settings.

Firstly, we collected training data for ANN using two distinct strategies for supervision, viz. CONAN [2] and Williams' [148]. Both were run for 500 simulations

[4] For each ZoA=H,A,L, we tested (MD=H vs MD=A) and (MD=A vs MD=L). For each MD=H,A,L, we tested (ZoA=L vs ZoA=A) and (ZoA=L vs ZoA=H).

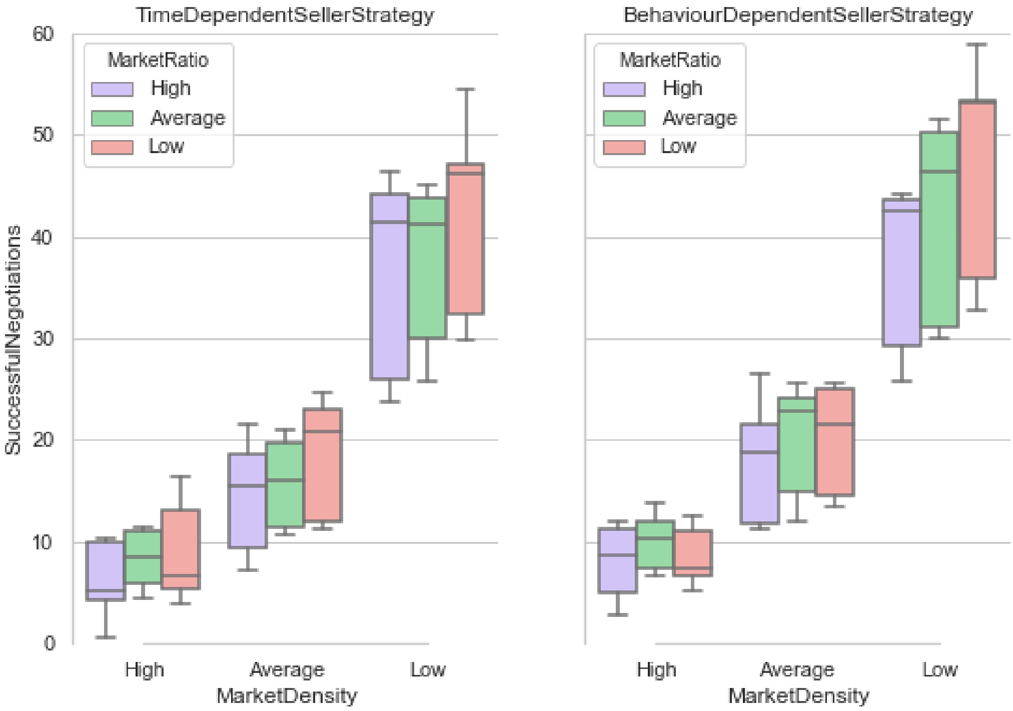

Figure 3.7: Effect of Market Density and Market Ratio on Percentage of Successful Negotiations

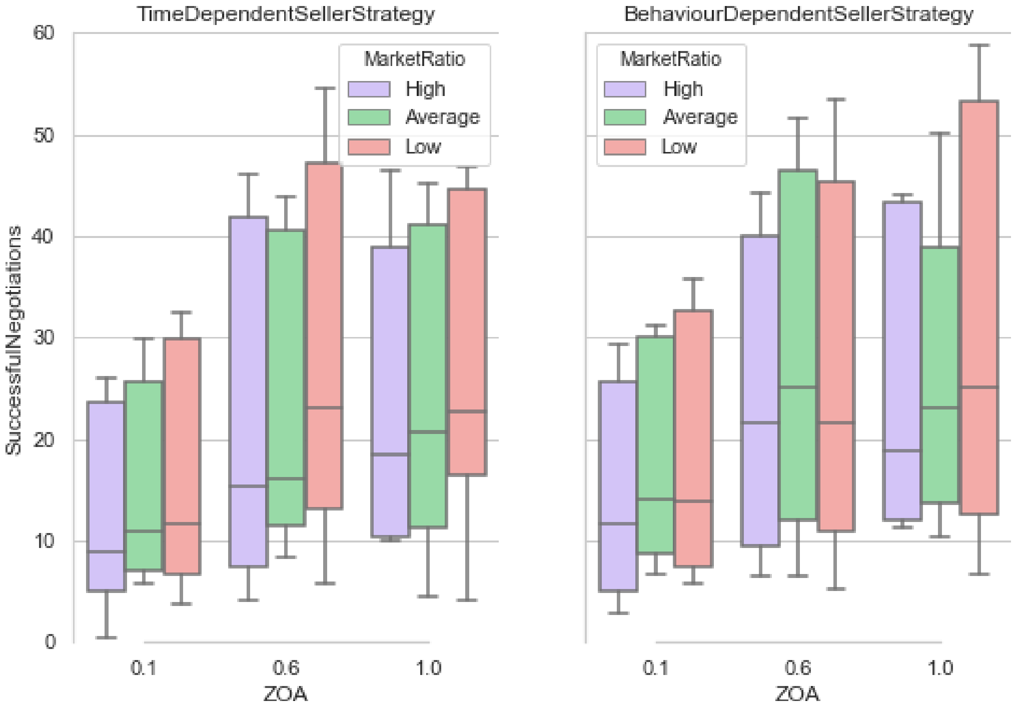

Figure 3.8: Effect of Market Ratio and ZoA on Percentage of Successful Negotiations

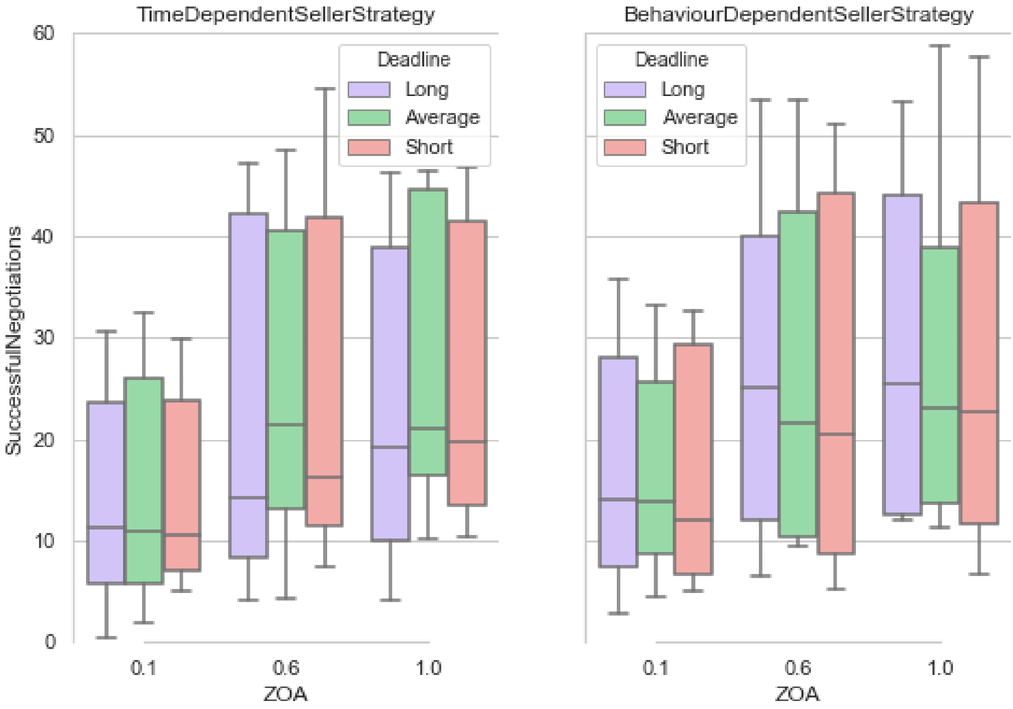

Figure 3.9: Effect of Deadline and ZoA on Percentage of Successful Negotiations

and with the same settings. Table 3.5 compares the performances of CONAN's and Williams' models. CONAN outperforms Williams' strategy in these settings in terms of U_{avg} and $S_{\%}$.

Then, the resulting trained ANN models (SL-C and SL-W) were used as the initial strategies in our DDPG-based DRL approach. These strategies evolved using negotiation experience from additional 500 simulations. In the remainder, we will abbreviate these trained models by *ANEGMA-C* and *ANEGMA-W* respectively.

Finally, we used test data from 101 simulations involving online learning to compare the performance of such derived *ANEGMA-C* and *ANEGMA-W* buyers against CONAN, Williams', SL-C, SL-W, and the so-called DRL model which used DDPG but initialized with a random strategy.

According to our results shown in Tables 3.7 and 3.8, the performance of SL-C is comparable to that of CONAN for both 60% and 100% *ZoA*s (see Table 3.5).

We observe the same for SL-W and the William's strategy. So, we conclude that our approach can successfully produce ANN strategies which are able to imitate the behaviour and performance of the CONAN and Williams' models (the training accuracies were in the range between 93.0% and 98.0% as shown in Table 3.6).

Even more importantly, the results demonstrate that *ANEGMA-C* (i.e., DDPG initialized with SL-C) and *ANEGMA-W* (i.e., DDPG initialized with SL-W) improve on their respective initial ANN strategies obtained by SL, and outperform DRL initialized at random for both 60% and 100% *ZoA*s, see Tables 3.7 and 3.8. This proves that both the evolution of the strategies via DRL and the initial supervision are beneficial. Furthermore, *ANEGMA-C* and *ANEGMA-W* also outperform the existing 'teacher strategies' (CONAN and Williams') in terms of U_{avg} used for the initial supervision and hence can improve on them, see Table 3.5.

Moving further, we observe that our agent ANEGMA becomes selective and learns to focus on how to obtain maximum utility from the end agreement (by accepting or proposing a bid only if a certain dynamic threshold utility is met). Thus, the successful negotiation rate is lower as compared to SL agents that seek to maximize the average utility rate. This could be a reason why SL-W seems to outperform ANEGMA-W in terms of successful negotiation rate. Although we could incorporate the number of successful negotiations in the reward function to bias our learning to optimize this metric, we have opted for the simple and commonly used reward function related to utility value only.

Table 3.5: Performance comparison of CONAN and Williams' model for both 60% and 100% *ZOA*. Best results are in bold.

Metric	CONAN		Williams'	
	Conceder Time Dependent Seller Strategy			
	60% ZoA	*100% ZoA*	*60% ZoA*	*100% ZoA*
U_{avg}	**0.27 ± 0.03**	**0.25 ± 0.07**	0.18 ± 0.08	0.17 ± 0.04
T_{avg}	**172942.78 ± 15177.77**	174611.43 ± 15139.52	177091.09 ± 15304.90	**174468.31 ±15365.11**
$S_{\%}$	**80.80**	**79.00**	78.20	78.00
	Relative Tit For Tat Behaviour Seller Strategy			
U_{avg}	**0.25 ± 0.03**	**0.24 ± 0.04**	0.22 ± 0.05	0.21 ± 0.06
T_{avg}	**176018.69 ± 14380.28**	179529.47 ± 14165.15	176334.65 ± 14683.03	**176468.31 ± 15365.11**
$S_{\%}$	**81.80**	**79.80**	73.00	73.21

Table 3.6: Training Accuracies of ANN (in %) when trained using datasets collected by negotiating CONAN (i.e., SL-C) and Williams' (i.e., SL-W) buyer strategy (for different *ZoA*s) against time-dependent and behaviour-dependent seller strategies

ZOA	Conceder Time Dependent		Relative Tit For Tat Behaviour Dependent	
	SL-C	SL-W	SL-C	SL-W
10%	93.88	94.73	94.65	94.77
60%	97.65	97.88	97.68	97.86
100%	95.96	96.23	96.88	95.73

Table 3.7: Performance comparison of SL VS ANEGMA VS DRL when *ZoA* is 60%. Best results are in bold. SL-C and SL-W correspond to ANN trained using data set collected from CONAN and Williams' approach respectively, whereas ANEGMA-C and ANEGMA-W correspond to DRL initialized with SL-C and SL-W respectively.

Metric	*SL-C*	*SL-W*	*ANEGMA-C*	*ANEGMA-W*	DRL
		Trained and Tested on Conceder Time Dependent Seller Strategy			
U_{avg}	0.27 ± 0.04	0.21 ± 0.08	**0.29 ± 0.04**	0.21 ± 0.04	-0.38 ± 0.14
T_{avg}	173529.47 ± 14651.15	171096.09 ± 14584.90	67750.62± 37628.57	132477.71 ± 26601.48	**768.55 ± 373.65**
$S_{\%}$	81.18	80.19	**87.12**	81.19	64.36
		Trained and Tested on Relative Tit for Tat Behaviour Dependent Seller Strategy			
U_{avg}	0.26 ± 0.03	0.23 ± 0.05	**0.29 ± 0.03**	0.23 ± 0.14	-0.19 ± 0.42
T_{avg}	167183.62 ± 13388.30	169334.65 ± 12389.03	36331.34 ± 70247.33	41225.17 ± 72938.79	**755.74 ± 292.29**
$S_{\%}$	82.18	75.24	**85.15**	74.26	61.38

Table 3.8: Performance comparison of SL VS ANEGMA VS DRL when *ZoA* is 100%. Best results are in bold. SL-C and SL-W correspond to ANN trained using data set collected from CONAN and Williams' approach respectively, whereas ANEGMA-C and ANEGMA-W correspond to DRL initialized with SL-C and SL-W respectively.

Metric	*SL-C*	*SL-W*	*ANEGMA-C*	*ANEGMA-W*	DRL
		Trained and Tested on Conceder Time Dependent Seller Strategy			
U_{avg}	0.23 ± 0.04	0.17 ± 0.08	**0.27 ± 0.51**	0.21 ± 0.71	-0.88 ± 0.16
T_{avg}	172234.73 ± 14516.15	170969.09 ± 14464.09	171266.64 ± 11573.38	185425.74 ± 19909.06	**1021.95 ± 771.47**
$S_{\%}$	77.23	76.24	**78.22**	72.28	57.42
		Trained and Tested on Relative Tit for Tat Behaviour Dependent Seller Strategy			
U_{avg}	0.26 ± 0.30	0.18 ± 0.55	**0.29 ± 0.35**	0.23 ± 0.84	-0.24 ± 0.55
T_{avg}	160178.98 ± 14809.18	163943.05 ± 12895.03	33695.16 ± 64292.37	23528.25 ± 61440.37	**817.67±523.67**
$S_{\%}$	73.27	72.28	**79.21**	71.81	56.43

Table 3.9: Performance comparison for the adaptive behaviour of SL VS ANEGMA VS DRL when ZoA is 60%. Best results are in bold. SL-C and SL-W correspond to ANN trained using data set collected from CONAN and Williams' approach respectively, whereas ANEGMA-C and ANEGMA-W correspond to DRL initialized with SL-C and SL-W respectively.

Metric	*SL-C*	*SL-W*	*ANEGMA-C*	*ANEGMA-W*	*DRL*
Trained on Relative Tit for Tat Behaviour Dependent and Tested on Conceder Time Dependent Seller Strategy					
U_{avg}	0.16 ± 0.05	0.17 ± 0.04	**0.26 ± 0.06**	0.23 ± 0.07	-0.36 ± 0.12
T_{avg}	174139.30 ± 14655.42	174035.91 ± 14627.59	38402.78± 64367.45	108051.11 ± 57755.84	**738.55 ± 279.65**
$S_{\%}$	70.29	69.30	**86.13**	81.19	54.45
Trained on Conceder Time Dependent and Tested on Relative Tit for Tat Behaviour Dependent Seller Strategy					
U_{avg}	0.25 ± 0.05	0.21 ± 0.04	**0.28 ± 0.01**	0.21 ± 0.08	-0.28 ± 0.51
T_{avg}	176048.05 ± 14423.36	175170.19 ± 14623.53	19295.84 ± 53767.54	114510.00 ± 64667.79	**806.83 ± 375.51**
$S_{\%}$	79.21	76.23	**84.16**	71.28	51.48

Hypothesis C *(ANEGMA is Adaptable)*

In this final test, we evaluate how well an *ANEGMA* agent can adapt to environments different from those used at training-time. Specifically, we deploy strategies trained using *Conceder Time Dependent* opponents into an environment with *Relative Tit for Tat Behaviour Dependent* opponents, and vice-versa. The *ANEGMA* agents use experience from 500 simulations to adapt to the new environment. Results are presented in Table 3.9 and show clear superiority of the *ANEGMA* agents over the SL-C and SL-W strategies which, without online retraining, cannot maintain their performance in the new environment. This confirms our hypothesis that *ANEGMA* agents can learn to adapt at run-time to different unknown seller strategies.

Further Discussion

Pondering over the negative average utility of DRL (see Tables 3.7 and 3.8), recall that we define utility as in Equation (3.6) but without the discount factor. Therefore, if an agent concedes a lot to make a deal, it will collect negative utility. This is precisely what happens to the initial random (and inefficient) strategy used in DRL. The combination of SL and DRL prevents this problem as it uses an initial pre-trained strategy which is much less likely to incur negative utility. For the same reason, we observe a consistently shorter T_{avg} for DRL caused by a buyer that concedes more to reach the agreement without negotiating for a long time with the seller. Hence, a shorter T_{avg} alone does not generally imply a better negotiation performance. An additional advantage of our approach is that it alleviates the common limitation of RL, namely, that an RL agent needs a non-trivial amount of experience before reaching satisfactory performance.

Results Summary

In this subsection, we summarize the results from Tables 3.7 to 3.9. When ZoA is 60% and 100%, *ANEGMA-C* outperforms all other strategies in comparison w.r.t U_{avg} and $S_{\%}$. However, DRL outperforms in terms of T_{avg}. We have also shown the results for the adaptive behaviour of *ANEGMA* when ZoA is 60%, which also reflects the same outcomes, i.e., *ANEGMA-C* outperforms all other agents in terms of average utility rate and number of successful negotiations.

Here, we conclude that the ANEGMA strategy becomes picky in that it learns to prefer settings where the agreements give high utility for the agent, even if in the tournament the number of successful agreements achieved by the agent so far is low. Moreover, ANEGMA supports single-issue negotiation where the issue has a range of real (continuous) values like price, or any other issues with values that can be mapped to intervals of real numbers. In this way, both negotiating parties will have their own initial and reservation values to negotiate with.

3.5 Summary

In this chapter, we proposed *ANEGMA*, a novel agent model for single-issue concurrent bilateral negotiation based on DRL and SL. In order to implement our model, we also extended the RECON simulation environment [1] to support agent learning during concurrent bilateral negotiation. Moreover, we performed rigorous experimental evaluations, demonstrating that *ANEGMA* outperforms the state-of-the-art in one-to-many concurrent bilateral negotiations. Furthermore, we observed that the *ANEGMA* agents can quickly adapt to a range of e-market settings. However, the *ANEGMA* approach is not interpretable and supports only single-issue bilateral negotiations. In the next chapter, we will discuss what changes are required to extend our current approach to support multi-issue negotiation and make it interpretable.

Chapter 4

Multiple-Issues Bilateral Negotiation Model - I

In the previous chapter, we discussed how to learn strategies for single-issue concurrent bilateral negotiation. In this chapter, we propose an agent model that provides learnable, adaptive and interpretable negotiation strategies for multiple issues under user and opponent preferences' uncertainty. We start off with the motivation of such an agent negotiation model in Section 4.1, and continue in Section 4.2 to describe the agent negotiation environment. Then, in Section 4.3, we present our newly proposed model *ANESIA*, including the idea of strategy templates and how they are used to deal with multiple issues. Afterwards, we throw a light upon various methods used along with the experimental settings in Section 4.4. In Section 4.5, we analyse the experimental results and compare the proposed model with the winning strategies of ANAC'17, '18 and '19 competitions using GENIUS. Finally, in Section 4.6, we present the summary of this chapter.

4.1 Motivation

We are interested in bilateral negotiations over domains with multiple issues. Negotiations in these domains are not necessarily concurrent, and may contain a possible outcome for each combination that can be formed from the values of each issue.

Consequently, such domains may have a large outcome space. In negotiations over such domains, finding a bid that is acceptable to both parties becomes more of a challenge than in a smaller domain. Another challenge in multi-issue negotiation is modelling a self-interested agent that learns to adapt its strategy while bilaterally negotiating against other agents. A model of this kind mostly considers the preferences of the user the agent represents in an application domain. However, users sometimes express their preferences by ranking only a few representative examples instead of providing a fully specified utility function [141]. This might be because (a) it is infeasible to ask a user to order or rank all outcomes when the outcome space is large; and (b) the user may have difficulty in assessing their preferences in a quantitative way. Thus, agents may be uncertain about the complete user preferences. Another challenge is the lack of knowledge about the preferences and negotiating characteristics of opponent agents [10].

A common assumption in automated negotiation is that normally there is no central entity as a mediator during the negotiation, so agents should find the solutions using a decentralized negotiation protocol. The closer our assumptions are to real-world applications, the more complicated negotiation settings we face, and more parameters are needed to design the negotiation strategy of the agent. Other assumptions that must be considered prior to designing an agent's negotiation strategy are as follows:

- We assume that the agents are bounded rational, i.e., their rationality or intelligence is limited while making decisions because of limited time or computational resources or information privacy.
- These agents have no previous knowledge of the preferences and negotiating characteristics of their opponents.
- The negotiation time is limited and there is a specific deadline for its termination, which means negotiating agents are under time pressure to reach an agreement,

therefore the agents must consider the risk of rejecting their offer from the opponent with regard to the limited time.

- The utility of offers might decrease over time (in negotiation scenarios with discount factor), thus, timely decision on rejecting or accepting an offer and making acceptable offers are of high importance for negotiators.

For such uncertain settings, one-size-fits-all negotiation strategies based on predefined heuristics or hand-crafted tactics that, by empirical evidence or domain knowledge, are known to work effectively are not suitable. This is because one rather seeks strategies that can be learned from interactions with the opponents and can be adapted to different negotiation domains. So, the question is how to develop strategies that address the complexity of multi-issue negotiation, including the time constraints and the preferences of the users.

In this context, we propose a model that builds on so-called *strategy templates*, i.e., parametric strategies that incorporate multiple negotiations tactics for the agent to choose from. A “strategy template” is described by a set of condition-action rules to be applied at different stages during the negotiation. Crucially, such templates require no assumptions from the agent developer as to which tactic to choose at which particular phase of the negotiation: starting from a template, we automatically learn, via stochastic search, the best combination of tactics (and values of possible tactics’ parameters) to use at any time during the negotiation. Another advantage is that, being logical combinations of individual tactics, the resulting strategies are interpretable and, thus, can be explained to the user. While the template parameters are learned before the negotiation begins, the proposed model is designed to enable online learning and adaptation as well.

4.2 Negotiation Environment

We assume that our negotiation environment E consists of two agents A_u and A_o negotiating with each other over some domain D. A domain D consists of n different independent issues, $D = (I_1, I_2, \dots I_n)$, with each issue taking a finite set of k possible discrete or continuous values $I_i = (v_1^i, \dots v_k^i)$. In our experiments, we consider issues with discrete values. An agent's bid ω is a mapping from each issue to a chosen value (denoted by c_i for the i-th issue), i.e., $\omega = (v_{c_1}^1, \dots v_{c_n}^n)$. The set of all possible bids or outcomes is called outcome space and is denoted by Ω s.t. $\omega \in \Omega$. The outcome space is common knowledge to the negotiating parties and stays fixed during a single negotiation session.

Negotiation protocol Before the agents can begin the negotiation and exchange bids, they must agree on a negotiation protocol P, which determines the valid moves agents can take at any state of the negotiation [47]. Here, we consider the alternating offers protocol [120], with possible $Actions = \{offer(\omega), accept, reject\}$. One of the agents (say A_u) starts a negotiation by making an offer $x_{A_u \to A_o}$ to the other agent (say A_o). The agent A_o can either accept or reject the offer. If it accepts, the negotiation ends with an agreement, otherwise A_o makes a counter-offer to A_u. This process of making offers continues until one of the agents either accepts an offer (i.e., successful negotiation) or the deadline is reached (i.e., failed negotiation).

Utility Each negotiating agent has certain preferences of how bids are offered over the other bids (i.e., cardinal preferences [95]), which is described by a preference profile. In contrast to the outcome space, the preference profile of the agent is private information. This profile is given in terms of a utility function U, defined as a weighted sum of evaluation functions $e_i(v_{c_i}^i)$ as expressed in (2.4) in Section 2.2.6.

Whenever the negotiation terminates without any agreement, each negotiating party gets its corresponding utility based on the private reservation value. Reservation value is defined as the minimum acceptable utility for an agent. Note that the reservation value may be different for different negotiation parties and also vary in different negotiation domains. In case the negotiation terminates with an agreement, each agent receives the discounted utility of the agreed bid, i.e., $U^d(\omega) = U(\omega)d_D^t$. Here, d_D is a discount factor in the interval $[0, 1]$ and $t \in [0, 1]$ is current normalized time.

User and opponent utility models Recall from Chapter 2 that our the negotiation environment is one with *incomplete information*, because the user utility model U_u is unknown. Also, estimating the user utility model $\widehat{U}_u$ from given partial preferences of the user leads to a single-objective constrained optimization problem, expressed as (2.5) in Section 2.2.6. In addition, we assume that our agent is unaware of the utility structure of its opponent agent U_o. Hence, to increase the agreement rate over multiple issues, our agent attempts to generate the (near) Pareto-optimal solutions during the negotiation which can be defined as a MOO problem, expressed as (2.6) in Section 2.2.6.

4.3 The *ANESIA* Model

In this section, we introduce *ANESIA* (*A*daptive *NE*gotiation model for a *S*elf-*I*nterested *A*utonomous agent) model and its components. Later, we also define the strategy templates used in the proposed model.

4.3.1 *ANESIA* Components

As shown in Figure 4.1, our agent A_u situates in an environment E, and interacts with another agent in the same environment. At any time t, our agent senses the current state (S_t) of E and represents it locally in the form of internal attributes. These include information derived from the sequence of previous bids offered by the opponent agent (e.g., utility of the best opponent bid so far, average utility of all the opponent bids and their variability) and information stored in our agent's

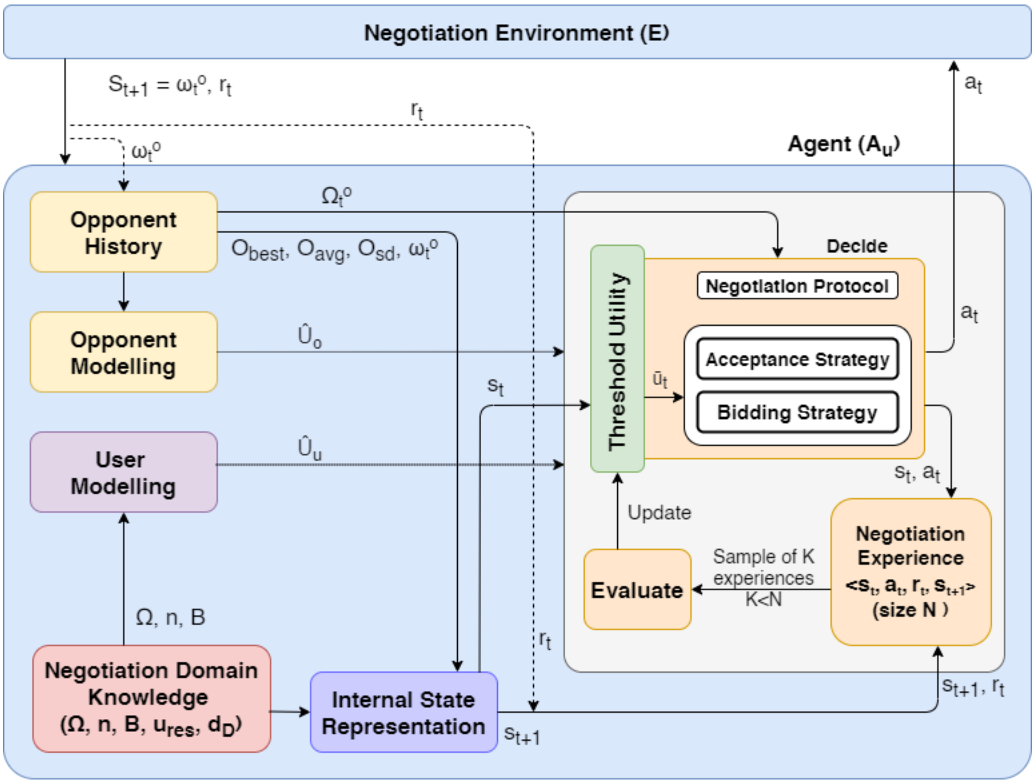

Figure 4.1: Interaction between the components of *ANESIA*

knowledge base (e.g., number of bids B in the given partial order, Ω, and n), and the current negotiation time t. This internal state representation, denoted with s_t, is used by the agent (in acceptance and bidding strategies) to decide what action a_t to execute. Action execution then changes the state of the environment to S_{t+1}.

As before with the *ANEGMA* model, learning in *ANESIA* mainly consists of three components: *Decide*, *Negotiation Experience*, and *Evaluate*. *Decide* refers to the negotiation strategy for choosing a near-optimal action a_t among a set of *Actions* at a particular state s_t based on a protocol P. Action a_t is derived via two functions, f_a and f_b, for the acceptance and bidding strategies, respectively. Function f_a takes as inputs s_t, a *dynamic threshold utility* $\bar{u}_t$ (defined later in the Methods section), the sequence of past opponent bids Ω_t^o, and outputs a discrete action a_t among *accept* or *reject*. When f_a returns *reject*, f_b computes what to bid next, with input s_t and $\bar{u}_t$, see (4.1–4.2). This separation of acceptance and bidding strategies is not rare, see for instance [11].

$$f_a(s_t, \bar{u}_t, \Omega_t^o) = a_t, a_t \in \{accept, reject\} \tag{4.1}$$

$$f_b(s_t, \bar{u}_t, \Omega_t^o) = a_t, a_t \in \{\mathit{offer}(\omega), \omega \in \Omega\} \tag{4.2}$$

Since we assume incomplete user and opponent preference information, *Decide* uses the estimated models $\widehat{U}_u$ and $\widehat{U}_o$. In particular, $\widehat{U}_u$ is estimated once before the negotiation starts by solving (2.5) and using the given partial preference profile $\preceq$. This encourages agent autonomy and avoids continuous user preference elicitation. Similarly, $\widehat{U}_o$ is estimated at time t using information from Ω_t^o, see Section 4.4 for more details.

Negotiation Experience stores historical information about N previous interactions (or experiences) of an agent with other agents. Experience elements are of the form $\langle s_t, a_t, r_t, s_{t+1} \rangle$, where s_t is the internal state of the negotiation environment E, a_t is an action performed by the agent at s_t, r_t is a scalar *reward* received from the environment and s_{t+1} is a new internal state after executing a_t.

Evaluate refers to a critic which helps our agent learn the dynamic threshold utility $\bar{u}_t$, which is evolved as new negotiation experience is collected. More specifically, it is a function of random K $(K < N)$ past negotiation experiences fetched from the agent's memory. The process of learning $\bar{u}_t$ is *retrospective* since it depends on the reward r_t obtained from E by performing action a_t at state s_t. The value of the reward depends on the (estimated) discounted utility of the last bid received from the opponent, ω_t^o, or of the bid accepted by either parties ω^{acc} and is defined as follows:

$$r_t = \begin{cases} \widehat{U}(\omega^{acc}, t), & \text{on agreement} \\ \widehat{U}(\omega_t^o, t), & \text{on received offer} \\ -1, & \text{otherwise} \end{cases} \tag{4.3}$$

where $\widehat{U}(\omega, t)$ is the discounted reward of ω defined as

$$\widehat{U}(\omega, t) = \widehat{U}(\omega) \cdot d^t, d \in [0, 1] \tag{4.4}$$

where d is a temporal discount factor included to encourage the agent to negotiate without delay. If $d = 1$, the utility is considered undiscounted. We should not confuse d, which is typically unknown to the agent, with the discount factor used to compute the utility of an agreed bid (d_D).

4.3.2 Specification of Strategy templates

ANESIA does not assume pre-defined strategies for f_a and f_b, and learns these strategies *offline*. To enable strategy learning, we introduce the notion of *strategy templates*, i.e., parametric strategies incorporating a series of tactics, where each tactic is executed for a specific negotiation phase. The parameters describing the start and duration of each phase, as well as the particular tactic choice for that phase are all *learnable* (blue-coloured in (4.5), (4.6)). Moreover, tactics can expose, in turn, learnable parameters themselves. We run multiple negotiations between our agent and a pool of opponents. We select the combination of tactics that maximizes the *true* user utility over these negotiations. So, in this stage only, we assume that the true user model is known.

We assume a collection of acceptance and bidding tactics, $\mathcal{T}_a$ and $\mathcal{T}_b$. Each $\mathtt{t}_a \in \mathcal{T}_a$ maps the agent state, threshold utility, opponent bid history, and a (possibly empty) vector of learnable parameters $\mathbf{p}$ into a utility value: if the agent is using tactic $\mathtt{t}_a$ and $\mathtt{t}_a(s_t, \bar{u}_t, \Omega^o_t, \mathbf{p}) = u$, then it will not accept any offer with utility below u, see (4.5) below. Each $\mathtt{t}_b \in \mathcal{T}_b$ is of the form $\mathtt{t}_b(s_t, \bar{u}_t, \Omega^o_t, \mathbf{p}) = \omega$ where $\omega \in \Omega$ is the bid returned by the tactic. An *acceptance strategy template* is a parametric function given by

$$\bigwedge_{i=1}^{n_a} t \in [t_i, t_{i+1}) \rightarrow \left(\bigwedge_{j=1}^{n_i} c_{i,j} \rightarrow \widehat{U}(\omega_t^o) \geq \mathtt{t}_{i,j}(s_t, \bar{u}_t, \Omega_t^o, \mathbf{p}_{i,j}) \right) \tag{4.5}$$

where n_a is the number of phases; $t_1 = 0$, $t_{n_a+1} = 1$, and $t_{i+1} = t_i + \delta_i$, where the δ_i parameter determines the duration of the i-th phase; for each phase i, the strategy template includes n_i tactics to choose from: $c_{i,j}$ is a Boolean choice parameter determining whether tactic $\mathtt{t}_{i,j} \in \mathcal{T}_a$ should be used during the i-th phase. We note that (4.5) is a predicate returning whether or not the opponent bid ω_t^o is accepted. Similarly, a *bidding strategy template* is defined by

$$\bigcup_{i=1}^{n_b} \begin{cases} \mathtt{t}_{i,1}(s_t, \bar{u}_t, \Omega_t^o, \mathbf{p}_{i,1}) & \text{if } t \in [t_i, t_{i+1}) \text{ and } c_{i,1} \\ \cdots & \cdots \\ \mathtt{t}_{i,n_i}(s_t, \bar{u}_t, \Omega_t^o, \mathbf{p}_{i,n_i}) & \text{if } t \in [t_i, t_{i+1}) \text{ and } c_{i,n} \end{cases} \tag{4.6}$$

where n_b is the number of phases, n_i is the number of options for the i-th phase, and $\mathtt{t}_{i,j} \in \mathcal{T}_b$. t_i and $c_{i,j}$ are defined as in the acceptance template. The particular libraries of tactics used in this work are discussed in the next Section. We stress that both (4.5) and (4.6) describe time-dependent strategies where a given choice of tactics is applied at different phases (denoted by the condition $t \in [t_i, t_{i+1})$).

4.4 Setting Up *ANESIA* for Experiments

User modelling: Before the negotiation begins, we estimate the user model $\widehat{U}_u$ by finding the weights w_i and utility values $e_i(v^i_{c_i})$ for each issue i, see (2.4), so that the resulting bid ordering best fits the given partial order $\preceq$ of bids. To solve this optimization problem (2.5), we use FA (Firefly Algorithm) [151], a meta-heuristic inspired by the swarming and flashing behaviour of tropical fireflies, because, in our preliminary analyses, it outperformed other traditional nature-inspired meta-heuristics such as GA and PSO [115]. In the FA metaphor, the candidate solution

$\widehat{U}'_u$ can be perceived as an agent (firefly) whose brightness depends on the objective function (or fitness value). The search space is explored by moving each firefly towards a brighter partner firefly in each iteration. After the maximum number of iterations, the brightest firefly (i.e., the one with maximum fitness value) is chosen as the best solution. We compute the fitness of a candidate solution (i.e., the user model $\widehat{U}'_u$) as the Spearman's rank correlation coefficient ρ between the estimated ranking of $\widehat{U}'_u$ and the true, but partial, bid ranking $\preceq$. The coefficient $\rho \in [-1, 1]$ is indeed a similarity measure between two rankings, assigning a value of 1 for identical and -1 for opposed rankings.

Opponent modelling: To derive an estimate of the opponent model $\widehat{U}_o$ during negotiation, we use the distribution-based frequency model proposed in [142]. In this model, the empirical frequency of the issue values in Ω^o_t provides an educated guess on the opponent's most preferred issue values. The issue weights are estimated by analysing the disjoint windows of Ω^o_t, giving an idea of the shift of opponent's preferences from its previous negotiation strategy over time.

Utility threshold learning: As before, we use an actor-critic architecture with model-free deep reinforcement learning (i.e., Deep Deterministic Policy Gradient (DDPG) [86]) to predict the target threshold utility $\bar{u}_t$. Also, as before, we consider a model-free RL approach because our problem is how to make an agent decide what target threshold utility to set next in a negotiation dialogue rather than predicting the new state of the environment, which implies model-based RL. Thus, $\bar{u}_t$ is expressed as a deep neural network function, which takes the agent state s_t as an input (see previous section for the list of attributes). Prior to RL, our agent's strategy is pre-trained with supervision from synthetic negotiation data. To collect supervision data, we use the *GENIUS* simulation environment [88], which supports multi-issue bilateral negotiation for different domains and user profiles. In particular, data was generated by running the winner of the ANAC'19 (AgentGG) against other strategies[1] in three different domains[2] and assuming no user preference uncertainties [6]. This initial supervised learning (SL) stage helps our agent decrease the exploration time required for DRL during the negotiation, an idea primarily influenced by the work of [16].

Strategy learning: The parameters of the acceptance and bidding strategy templates (4.5–4.6) are learned by running the FA meta-heuristic. We define the fitness of a particular choice of template parameters as the average *true* user utility over multiple negotiations rounds under the concrete strategy implied by those parameters. Negotiation data is obtained by running our agent on the GENIUS platform against three (readily available) opponents (*AgentGG*, *KakeSoba* and *SAGA*) in three different negotiation domains[2].

We now describe the libraries of tactics (with learnable parameters in blue color to distinguish them from non-learnable parameters) used in our templates. For the acceptance tactics, we consider:

- $\widehat{U}_u(\omega_t)$, the estimated utility of the bid that our agent would propose at the time t ($\omega_t = f_b(s_t, \bar{u}_t, \Omega^o_t)$).
- $Q_{\widehat{U}_u(\Omega^o_t)}(a \cdot t + b)$, where $\widehat{U}_u(\Omega^o_t)$ is the distribution of (estimated) utility values of the bids in Ω^o_t, $Q_{\widehat{U}_u(B_o(t))}(p)$ is the quantile function of such distribution, and a and b are learnable parameters. In other words, we consider the p-th best utility received from the agent, where p is a learnable (linear) function of the negotiation time t. In this way, this tactic automatically and dynamically decides how much the agent should concede at time t.
- $\bar{u}_t$, the dynamic DRL-based utility threshold.
- $\bar{u}$, a fixed, but learnable, utility threshold.

The bidding tactics in our library are:

[1] *Gravity*, *HardDealer*, *Kagent*, *Kakesoba*, *SAGA*, and *SACRA*.

[2] Laptop, Holiday and Party.

- $b_{Boulware}$, a bid generated by a time-dependent Boulware strategy [43].

- $PS(a \cdot t + b)$ extracts a bid from the set of Pareto-optimal bids PS (see (2.7)), derived using the *NSGA-II algorithm*[3] [39] under $\widehat{U}_u$ and $\widehat{U}_o$. In particular, it selects the bid that assigns a weight of $a \cdot t + b$ to our agent utility (and $1 - (a \cdot t + b)$ to the opponent's), where a and b are learnable parameters telling how this weight scales with the negotiation time t. The *TOPSIS algorithm* [64] is used to derive such a bid, given the weighting $a \cdot t + b$ as input.

- $b_{opp}(\omega_t^o)$, a tactic to generate a bid by manipulating the last bid received from the opponent ω_t^o. This is modified in a greedy fashion by randomly changing the value of the least relevant issue (w.r.t. $\widehat{U}$) of ω_t^o.

- $\omega \sim \mathcal{U}(\Omega_{\geq \bar{u}_t})$, a random bid above our DRL-based utility threshold $\bar{u}_t$[4].

Below we give an example of a concrete acceptance strategy learned in our experiments: it employs time-dependent quantile tactic during the middle of the negotiation, and the DRL threshold utility during the initial and final stages.

$$
\begin{aligned}
t \in [0.0, 0.4) &\to \widehat{U}(\omega_t^o) \geq \bar{u}_t \wedge \bar{u} \\
t \in [0.4, 0.7) &\to \widehat{U}(\omega_t^o) \geq \widehat{U}(\omega_t) \wedge Q_{\widehat{U}(\Omega_t^o)}(-0.67 \cdot t + 1.27) \\
t \in [0.7, 0.95) &\to \widehat{U}(\omega_t^o) \geq \widehat{U}(\omega_t) \wedge Q_{\widehat{U}(\Omega_t^o)}(-0.21 \cdot t + 0.9) \\
t \in [0.95, 1.0] &\to \widehat{U}(\omega_t^o) \geq \bar{u}_t
\end{aligned}
$$

Below is an example of a learned concrete bidding strategy: it behaves in a Boulware-like manner in the initial stage, after which it proposes near Pareto-optimal bids

[3] Meta-heuristics (instead of brute-force) for Pareto-optimal solutions have the potential to deal efficiently with continuous issues.

[4] $\mathcal{U}(S)$ is the uniform distribution over S, and $\Omega_{\geq \bar{u}_t}$ is the subset of Ω whose bids have estimated utility above $\bar{u}_t$ w.r.t. $\widehat{U}$.

(between time 0.4 and 0.9) and opponent-oriented bid in the final stage.

$$
\begin{aligned}
t \in [0.0, 0.4) &\rightarrow \omega = b_{Boulware} \\
t \in [0.4, 0.9) &\rightarrow \omega = PS(-0.75 \cdot t + 0.6) \\
t \in [0.9, 1.0] &\rightarrow \omega = b_{opp}(\omega_t^o)
\end{aligned}
$$

We stress that our approach allows to automatically devise such combinations of tactics so as to achieve optimal user utility, which would be infeasible manually.

NSGA-II: In order to contribute to more "win-win" negotiation agreements, our agent attempts to offer Pareto-optimal bids during the negotiation. This Pareto front represents a solution of a multi-objective optimization problem (i.e., maximizing our agent's utility as well as opponent agent's utility) expressed as (2.6), dealing with multiple conflicting objectives (i.e., increasing one agent's utility decreases the other). In our model, our agent uses a well-known fast non-dominated sorting evolutionary algorithm known as NSGA-II [39] which has mainly three important features: *elitism* (few individuals from the population are given opportunity to move to the next generation), *crowding distance* for diversity preserving and emphasising the *non-dominated solutions*. In our model, one individual, say A(x_1, y_1), dominates other, say B(x_2, y_2), if $x_1 \geq x_2$ and $y_1 \geq y_2$ and $(x1 > x2$ or $y1 > y2))$. Here, x_i is a utility value obtained by applying x objective function on individual i. For more details on the implementation of NSGA-II, see [39].

TOPSIS: During the bidding phase, NSGA-II generates a Pareto-frontier which may contain more than one bid. In order to decide one bid among n alternatives/choices, our agent uses a Multi-Criteria Decision-Making (MCDM) method called TOPSIS (Technique for Order Preference by Similarity to the Ideal Solution) [64]. In our model, we have only two criteria ($m = 2$) or objectives: maximizing user utility and maximizing opponent utility (since our focus is on more win-win situations), based on which n alternatives will be ordered. Our agent implements

TOPSIS as follows:

- A decision matrix $M = n \times m$ consists of n alternatives and m criteria is created. We assume $m = 2$ i.e., m_1 and m_2, where $m_1 = \widehat{U}(\omega_i)$ and $m_2 = \widehat{U}(\omega_i^o)$.

- The next step is normalizing the decision matrix M using (4.7), where $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$ and x_{ij} is a value assigned to the i^{th} alternative w.r.t j^{th} criteria.

$$r_{ij} = \frac{x_{ij}}{\left[\sum_{k=1}^{n}(x_{kj})^2\right]^{1/2}} \tag{4.7}$$

- Next step is to create a weighted normalized decision matrix W using where x_{ij} is replaced with v_{ij} and $v_{ij} = w_j \cdot r_{ij}$. In our model, w_j are learnable parameters which tells how these weights scale with negotiation time t. So, $w_1 = a \cdot t + b$ and $w_2 = 1 - (a \cdot t + b)$. From the example given in the previous section of strategy template, $a = -0.75$ and $b = 0.6$.

- Once the weighted normalized matrix is ready, the distance of each alternative from an ideal positive and ideal negative solutions is computed.

- Finally, the ranks are ordered from high to bottom based on the relative closeness of each alternative to the ideal solutions.

An alternative with top rank is chosen by our agent to propose to the opponent agent during this time period.

4.5 Experimental Setup and Results

4.5.1 Experimental Hypotheses

All the experiments have been performed using the GENIUS negotiation platform [88]. Our experiments are designed to prove the following hypotheses:

Hypothesis A: Our approach can well approximate user models under user preference uncertainty.

Hypothesis B: The set of NSGA-II estimated Pareto-optimal bids are close to the true Pareto-optimal front.

Hypothesis C: *ANESIA* outperforms the “teacher” strategies (AgentGG, Kake-Soba and SAGA) in known negotiation settings in terms of individual and social efficiency.

Hypothesis D: *ANESIA* outperforms not-seen-before negotiation strategies and adapts to different negotiation settings in terms of individual and social efficiency.

4.5.2 Performance Metrics

We measure the performance of each agent in terms of six widely-adopted metrics inspired by the ANAC competition:

- U_{ind}^{total}: The utility gained by an agent averaged over all the negotiations ($\uparrow$);
- U_{ind}^{s}: The utility gained by an agent averaged over all the *successful* negotiations ($\uparrow$);
- U_{soc}: The utility gained by both negotiating agents averaged over all successful negotiations ($\uparrow$);
- P_{avg}: Average minimal distance of agreements from the Pareto Frontier ($\downarrow$).
- R_{avg}: Average number of rounds before reaching the agreement ($\downarrow$);
- $S_{\%}$: Proportion of successful negotiations ($\uparrow$).

The first and second measures represent *individual efficiency* of an outcome, whereas the third and fourth correspond to the *social efficiency* of agreements.

4.5.3 Experimental Settings

ANESIA is evaluated against state-of-the-art strategies that participated in ANAC’17, ’18, and ’19, and designed by different research groups independently. Each agent has no information about another agent’s strategies beforehand. Details of all these strategies are available in [7, 72, 6]. We assume incomplete information about user preferences, given in the form of B randomly-chosen partially-ordered bids. We evaluate *ANESIA* on 8 negotiation domains (see Appendix A) which are different from each other in terms of size and opposition [8] to ensure good negotiation characteristics and to reduce any biases. The domain size refers to the number of issues, whereas opposition[5] refers to the minimum distance from all possible outcomes to the point representing complete satisfaction of both negotiation parties (1,1). For our experiments, we choose readily-available 3 small-sized, 2 medium-sized, and 3 large-sized domains. Out of these domains, 2 are with high, 3 with medium and 3 with low opposition (see [149] for more details).

For each configuration, each agent plays both roles in the negotiation to compensate for any utility differences in the preference profiles. We call *user profile* the agent's role along with the user's preferences. We set two user preferences uncertainties for each role: $|B| = 5\%|\Omega|$ and $|B| = 10\%|\Omega|$. Also, we set the u_{res} and d_D to their respective default values, whereas the deadline is set to 60s, normalized in $[0, 1]$ (known to both negotiating parties in advance).

Regarding the optimization algorithms, for FA (hypotheses A and C), we choose a population size of 20 and 200 generations for user model estimation and learning of strategy template parameters. We also set the maximum attractiveness value to 1.0 and absorption coefficient to 0.01. For NSGA-II (hypothesis B), we choose the population size of $2\% \times |\Omega|$, 2 generations and mutation count of 0.1. With these hyperparameters, on our machine[6] the run-time of NSGA-II never exceeded the given timeout of 10s for deciding an action at each turn, while being able to retrieve empirically good solutions. We choose these hyper-parameters so that the run-time of the algorithms do not exceed the given timeout of 10s for deciding an

[5] The value of opposition reflects the competitiveness between parties in the domain. Strong opposition means a gain of one party is at the loss of the other, whereas, weak opposition means that both parties either lose or gain simultaneously [8].

[6] CPU: 8 Cores, 2.10GHz; RAM: 32GB

action at each turn, while being able to retrieve empirically good solutions.

4.5.4 Empirical Evaluation

Hypothesis A: User Modelling

We used two measures to determine the difference between $\widehat{U}_u$ and U_u [144]: First, *Ordinal accuracy (OA)* measures the proportion of bids put by $\widehat{U}$ in the correct rank order (i.e., as defined by the true user model), where an OA value of 1 implies a 100% correct ranking and verifies whether the estimated user model preserves the rank order for all issues and its values in domain D as defined by the true user model. It can be defined as the ratio of the number of elements of D for which the rank order in estimated and true user models are concordant (i.e., n^{con} to the number of all elements (n+1) defined in the negotiation domain D, as shown in (4.8).

$$OA = \frac{n^{con}}{n+1} \tag{4.8}$$

Second, to capture the scale of cardinal errors, *Cardinal Inaccuracy (CI)* measures the differences in ratings assigned in the estimated and true user models for all the elements in domain D. It can be defined as a multi-dimensional distance formula that determines the differences between the n various issue ratings and the value ratings for each issue j in estimated and true user models, as shown in (4.9) and (4.10) respectively.

$$II = \sum_{j=1}^{n} |w_j^{true} - w_j^{est}| \tag{4.9}$$

$$OI_j = w_j^{true}. \sum_{k=1}^{n_j} |\bar{v}^{true}(x_k^j) - \bar{v}^{est}(x_k^j)| \tag{4.10}$$

Finally, the CI index is defined as the sum of II and OI_j determined for each issue $j \in 1, 2, \ldots, n$, as shown in (4.11).

$$CI = II + \sum_{j=1}^{n} OI_j \tag{4.11}$$

We produced results in 8 domains and two profiles (5% and 10% of total possible bids) which are averaged over 10 simulations as shown in Table 4.1. All the values of OA (↑) and CI (↓), in each domain, for both the user profiles, are ≥ 0.67 and ≤ 0.90 respectively, which is quite accurate given the uncertainty and the fact that the CI value $\propto |D|$. We observed that ordinal and cardinal accuracies for 10% of Ω are higher compared to 5% of Ω, which is desirable. It is also interesting to note that within the ordinal accuracies (5% of Ω), for small values of $|B|$ (e.g., Flight), the accuracies are higher from large values of $|B|$ (e.g., Fitness). We believe this happens because of the randomness involved in a meta-heuristic approach where for large size of $|B|$ (e.g., Fitness), the uncertainty due to the combination of values in the issues is more difficult to approximate in absolute terms. In other words, the probability of making a mistake is higher in the presence of given ranking of more bids as compared to the domains with small size of $|B|$ where choices are very less.

Domain ($n, \|\Omega\|$)	Ordinal Accuracy (↑)		Cardinal Inaccuracy (↓)	
$\|B\|$	5% of $\|\Omega\|$	10% of $\|\Omega\|$	5% of $\|\Omega\|$	10% of $\|\Omega\|$
AirportSite (3, 420)	(0.75,0.75)	(0.85,0.87)	(0.47,0.54)	(0.78,0.76)
Camera (6, 3600)	(0.77,0.63)	(0.83,0.75)	(0.32,0.32)	(0.69,0.41)
Energy (6, 15625)	(0.74,0.78)	(0.83,0.84)	(0.56,0.61)	(0.57,0.69)
Fitness (5, 3520)	(0.67,0.67)	(0.70,0.75)	(0.55,0.47)	(0.46,0.59)
Flight (3, 48)	(0.75,0.85)	(0.82,0.90)	(0.65,0.75)	(0.58,0.79)
Grocery (5, 1600)	(0.67,0.67)	(0.75,0.72)	(0.35,0.42)	(0.39,0.56)
Itex-Cypress (4, 180)	(0.70,0.74)	(0.78,0.80)	(0.56,0.48)	(0.74,0.56)
Outfit (4, 128)	(0.70,0.75)	(0.80,0.84)	(0.71,0.88)	(0.89,0.79)

Table 4.1: Evaluation of User Modelling using FA for two profiles (separated by comma) in each domain

Hypothesis B: Pareto-Optimal Bids

We used a popular metric called Inverted Generational Distance (IGD) [36] to compare the Pareto Fronts found by the NSGA-II and the ground truth (found via brute force), and is defined as:

$$IGD(P_{true}, P_{est}) = \sum_{v \in P_{true}} \frac{d(v, P_{est})}{|P_{true}|} \tag{4.12}$$

In (4.12), P_{true} is a set of Pareto optimal bids in true Pareto Frontier, P_{est} is a set of approximation of optimal bids in true Pareto Frontier, $d(v, P_{est})$ is the minimum Euclidean distance between v and all the points in set P_{est}; and $|P_{true}|$ is the cardinality of set P_{true}. Small IGD values suggest good convergence of solutions to the Pareto Front and their good distribution over the entire Pareto Front. Table 4.2 demonstrates the potential of NSGA-II[7] for generating the Pareto-optimal bids as well as the closeness of true utility models.

Domain	IGD (↓)	Domain	IGD (↓)
Airport Site ($|\Omega| = 420$)	0.000	Flight ($|\Omega| = 48$)	0.006
Camera ($|\Omega| = 3600$)	0.000	Grocery ($|\Omega| = 1600$)	0.000
Energy ($|\Omega| = 15625$)	0.011	Itex-Cypress ($|\Omega| = 180$)	0.000
Fitness ($|\Omega| = 3520$)	0.012	Outfit ($|\Omega| = 128$)	0.000

Table 4.2: Evaluation of Pareto Frontier using Inverted Generational Distance estimated using NSGA-II

Hypothesis C: ***ANESIA*** outperforms "teacher" strategies

We performed a total of 1440 negotiation sessions[8] to evaluate the performance of *ANESIA* against the three "teacher" strategies (AgentGG, KakeSoba and SAGA) in three domains (Laptop, Holiday, and Party) for two different profiles ($|B| = 5, 10$). These strategies were used to collect the dataset in the same domains for supervised training before the DRL process begins. Table 4.3 demonstrates the average results over all the domains and profiles for each agent. Clearly, *ANESIA* outperforms the "teacher" strategies in terms of U^s_{ind} (i.e., individual efficiency), U_{soc}, and P_{avg} (i.e., social efficiency).

[7] Population size = $0.02 \times |\Omega|$; Number of generations = 2.

[8] $n \times (n-1)/2 \times x \times y \times z \times w = 1440$ where $n = 4$, number of agents in a tournament; $x = 2$, because agents play both sides; $y = 3$, number of domains; $z = 20$, because each tournament is repeated 20 times; $w = 2$, number of profiles in terms of B. We consider tournament settings to address some practical deep learning considerations, e.g., catastrophic forgetting, which means tendency of abruptly losing knowledge gained while negotiating against old opponent agent as information relevant to the current new opponent agent is incorporated [49].

Hypothesis D: Adaptive Behaviour of *ANESIA* agent

We further evaluated the performance of *ANESIA* on agents (from ANAC'17, ANAC'18 and ANAC'19) unseen during training. For this, we performed a total of 23040 negotiation sessions[9]. Results in Table 4.5(A) are averaged over all domains and profiles, and demonstrate that *ANESIA* learns to make the optimal choice of tactics to be used at run time and outperforms the other 8 strategies in terms of U^s_{ind} and U_{soc}. See Tables B.17 – B.24 in Appendix for results in separate domains and profiles.

Ablation Study 1: We evaluated the ANESIA-DRL performance, i.e., an *ANESIA* agent that does not use templates to learn optimal combinations of tactics, but uses only one acceptance tactic, given by the dynamic DRL-based threshold utility $\bar{u}_t$ (and the Boulware and Pareto-optimal tactics for bidding) for the same negotiation settings of Hypothesis D. We observe from Table 4.5(B) that *ANESIA-DRL* outperforms the other strategies in terms of U^s_{ind} and U_{soc}. See Tables B.9 – B.16 in Appendix for results in separate domains and profiles.

Ablation Study 2: We evaluated the ANESIA-rand performance, i.e., an *ANESIA* agent which starts from a random DRL policy, i.e., without any offline pre-training of the adaptive utility threshold $\bar{u}_t$, for the same negotiation settings of Hypothesis D. From Table 4.5(B), we observe in ANESIA-rand some degradation of the utility metrics compared to the fully-fledged *ANESIA* and ANESIA-DRL, even though it remains equally competitive w.r.t. the ANAC'17,18 agents and outperforms the ANAC'19 agents in P_{avg}, U^s_{ind} and U_{soc}. This is not unexpected because, with a poorly informed (random) target utility tactic, the agent tends to accept offers with little pay-off without negotiating for more rounds. See Tables B.1 – B.8 in Appendix for results in separate domains and profiles. From the Table 4.5(B) results, we conclude that the combination of both features (strategy templates and pre-training of the DRL model) are beneficial, even though these two features perform well in isolation too.

[9] $n \times (n-1)/2 \times x \times y \times z \times w = 23040$ where $n = 9$; $x = 2$; $y = 8$; $z = 20$; and $w = 2$.

Metric	ANESIA	AgentGG	KakeSoba	SAGA
$R_{avg}(\downarrow)$	301.64 ± 450.60	1327.51 ± 2246.83	1154.83 ± 2108.11	**287.34 ± 1058.52**
$P_{avg}(\downarrow)$	**0.12 ± 0.36**	0.30 ± 0.37	0.22 ± 0.34	0.18 ± 0.31
$U_{soc}(\uparrow)$	**1.55 ± 0.73**	1.10 ± 0.67	1.26 ± 0.62	1.36 ± 0.56
$U_{ind}^{total}(\uparrow)$	0.39 ± 0.41	0.65 ± 0.39	**0.71 ± 0.35**	0.58 ± 0.26
$U_{ind}^{s}(\uparrow)$	**0.92 ± 0.11**	0.87 ± 0.13	0.88 ± 0.12	0.67 ± 0.14
$S_{\%}(\uparrow)$	0.46	0.75	0.81	**0.87**

Table 4.3: Performance Comparison of *ANESIA* with “teacher” strategies. Results are averaged for all the domains and profiles. See supplement for separate results for each domain.

Metric	ANESIA		AgentGG		KakeSoba		SAGA	
	$\|B\| = 5$	$\|B\| = 10$	$\|B\| = 5$	$\|B\| = 10$	$\|B\| = 5$	$\|B\| = 10$	$\|B\| = 5$	$\|B\| = 10$
	Holiday Domain							
R_{avg}	476.81 ± 847.48	259.35 ± 453.3	1224.04 ± 1770.48	532.77 ± 841.16	1066.18 ± 1710.8	414.51 ± 719.58	32.3 ± 83.79	238.18 ± 783.88
P_{avg}	**0.15 ± 0.57**	**0.17 ± 0.26**	0.4 ± 0.46	0.35 ± 0.44	0.28 ± 0.44	0.17 ± 0.34	0.23 ± 0.37	0.31 ± 0.42
U_{soc}	**1.79 ± 0.81**	**1.72 ± 0.78**	1.25 ±0.66	1.33 ± 0.62	1.41 ± 0.63	1.57 ± 0.49	1.49 ± 0.53	1.39 ± 0.59
U_{ind}	0.46 ± 0.42	0.51 ± 0.39	0.67 ±0.37	0.73 ±0.34	0.73 ± 0.34	0.8 ± 0.26	0.65 ± 0.24	0.61 ± 0.29
U^s_{ind}	**0.88 ± 0.11**	**0.88 ± 0.01**	0.86 ±0.13	0.88 ±0.07	0.87 ± 0.13	0.87 ± 0.11	0.73 ± 0.1	0.71 ± 0.17
P_{succ}	0.56	0.64	0.79	0.83	0.84	**0.92**	0.89	0.86
	Laptop Domain							
R_{avg}	256.27 ± 393.1	254.62 ±446.96	2840.6 ± 5055.29	2186.61 ± 4523.85	2292.48 ± 4559.91	2054.73 ± 4445.48	875.89 ± 3305.07	232.84 ± 1316.79
P_{avg}	**0.02 ±0.26**	**0.05 ± 0.26**	0.14 ±0.23	0.13 ± 0.22	0.07 ± 0.18	0.07 ± 0.18	0.06 ± 0.16	0.06 ± 0.17
U_{soc}	**1.57 ±0.81**	**1.50 ± 0.8**	0.99 ± 0.71	1.07 ± 0.7	1.26 ± 0.63	1.34 ± 0.62	1.45 ± 0.56	1.48 ± 0.56
U_{ind}	0.47 ± 0.48	0.5 ± 0.48	0.68 ±0.42	0.72 ± 0.41	0.78 ± 0.33	0.77 ± 0.32	0.71 ± 0.26	0.73 ± 0.28
U^s_{ind}	**0.96 ± 0.08**	**0.95 ± 0.09**	0.93 ± 0.1	0.94 ± 0.09	0.91 ± 0.11	0.89 ± 0.1	0.8 ± 0.07	0.82 ± 0.09
P_{succ}	0.48	0.52	0.73	0.76	0.86	0.87	0.89	**0.89**
	Party Domain							
T_{avg}	47.24 ± 25.46	54.67 ± 40.18	38.85± 33.71	44.35± 27.26	41.19 ± 32.82	45.15 ± 26.6 7	16.11 ± 23.87	23.58 ± 44.59
R_{avg}	213.2 ± 274.16	229.98 ± 288.62	508.04± 607.31	673.01± 682.91	473.42 ± 560.71	627.69 ± 652.17	135.17 ± 347.58	209.67 ± 514.01
P_{avg}	**0.15 ±0.42**	**0.18 ±0.4**	0.39± 0.43	0.37± 0.46	0.4 ± 0.46	0.36 ± 0.46	0.21 ± 0.35	0.22 ± 0.39
U_{soc}	**1.41 ± 0.6**	**1.34 ± 0.59**	0.97± 0.64	1.0± 0.69	0.96 ± 0.69	1.0 ± 0.69	1.18 ± 0.51	1.17 ± 0.58
U_{ind}	0.23 ±0.35	0.2 ±0.35	0.57± 0.4	0.54± 0.41	0.56 ± 0.42	0.62 ± 0.43	0.42 ± 0.25	0.39 ± 0.26
U^s_{ind}	**0.92 ± 0.2**	**0.92 ± 0.17**	0.81 ± 0.17	0.79 ± 0.22	0.82 ± 0.19	0.91 ± 0.11	0.5 ± 0.2	0.48 ± 0.19
P_{succ}	0.32	0.26	0.7	0.69	0.67	0.69	**0.85**	0.82

Table 4.4: Performance Comparison of *ANESIA* with “teacher” strategies in three different domains, each having two different uncertain profiles $|B| = 5$ and $|B| = 10$

(A) Performance Analysis of fully-fledged ANESIA

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	626.24 ± 432.45	0.17 ± 0.29	**1.51 ± 0.47**	0.66 ± 0.25	**0.95 ± 0.06**	0.51
AgentGP ●	1366.95 ± 1378.19	0.30 ± 0.29	0.82 ± 0.58	0.66 ± 0.22	0.88 ± 0.09	0.58
FSEGA2019 ●	1801.96 ± 1754.91	0.17 ± 0.19	1.15 ± 0.37	0.74 ± 0.18	0.82 ± 0.11	0.83
AgentHerb ◇	**32.30 ± 50.53**	**0.01 ± 0.03**	1.41 ± 0.10	0.45 ± 0.13	0.45 ± 0.13	**1.00**
Agent33 ◇	4044.47 ± 4095.57	0.07 ± 0.16	1.31 ± 0.32	0.62 ± 0.15	0.64 ± 0.14	0.93
Sontag ◇	5129.47 ± 5855.11	0.10 ± 0.18	1.22 ± 0.37	0.73 ± 0.17	0.79 ± 0.11	0.86
AgreeableAgent ◇	5003.98 ± 5216.68	0.11 ± 0.23	0.9 ± 0.4	0.67 ± 0.20	0.73 ± 0.13	0.73
PonpokoAgent ⋆	6609.73 ± 6611.18	0.18 ± 0.26	1.01 ± 0.49	**0.76 ± 0.22**	0.89 ± 0.07	0.74
ParsCat2 ⋆	4971.43 ± 4911.8	0.11 ± 0.22	1.16 ± 0.42	0.75 ± 0.19	0.81 ± 0.12	0.85

(B) Performance Analysis of ANESIA-DRL (Ablation study 1)

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	480.43 ± 418.81	0.10 ± 0.27	**1.34 ± 0.51**	0.65 ± 0.22	**0.87 ± 0.14**	0.56
AgentGP ●	296.58 ± 340.15	0.24 ± 0.25	0.95 ± 0.48	0.61 ± 0.20	0.69 ± 0.15	0.75
FSEGA2019 ●	1488.63 ± 1924.03	0.22 ± 0.20	0.97 ± 0.39	0.67 ± 0.20	0.79 ± 0.12	0.77
AgentHerb ◇	**38.25 ± 79.16**	**0.09 ± 0.09**	1.33 ± 0.15	0.48 ± 0.16	0.48 ± 0.16	**0.99**
Agent33 ◇	2799.6 ± 3702.85	0.13 ± 0.15	1.22 ± 0.29	0.58 ± 0.15	0.59 ± 0.15	0.94
Sontag ◇	3490.01 ± 4909.81	0.17 ± 0.19	1.09 ± 0.37	0.69 ± 0.18	0.76 ± 0.13	0.85
AgreeableAgent ◇	3585.94 ± 4474.06	0.18 ± 0.21	0.78 ± 0.38	0.61 ± 0.19	0.68 ± 0.14	0.69
PonpokoAgent ⋆	4470.86 ± 5482.31	0.25 ± 0.24	0.90 ± 0.45	**0.71 ± 0.20**	0.85 ± 0.09	0.72
ParsCat2 ⋆	3430.55 ± 4204.93	0.17 ± 0.20	1.07 ± 0.39	0.69 ± 0.18	0.73 ± 0.16	0.85

(C) Performance Analysis of Random ANESIA (Ablation study 2)

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	351.77 ± 343.24	0.17 ± 0.21	1.28 ± 0.43	0.67 ± 0.22	0.74 ± 0.16	0.80
AgentGP ●	258.12 ± 300.53	0.22 ± 0.24	1.0 ± 0.46	0.63 ± 0.19	0.70 ± 0.14	0.79
FSEGA2019 ●	1002.3 ± 1349.64	0.20 ± 0.19	1.02 ± 0.35	0.68 ± 0.2	0.79 ± 0.12	0.81
AgentHerb ◇	**44.87 ± 80.16**	**0.09 ± 0.09**	**1.33 ± 0.15**	0.48 ± 0.15	0.48 ± 0.15	**0.99**
Agent33 ◇	1592.20 ± 2298.71	0.11 ± 0.12	1.26 ± 0.24	0.58 ± 0.15	0.58 ± 0.15	0.97
Sontag ◇	2276.96 ± 3366.20	0.15 ± 0.17	1.13 ± 0.34	0.70 ± 0.17	0.75 ± 0.13	0.88
AgreeableAgent ◇	2472.87 ± 3145.20	0.14 ± 0.17	0.86 ± 0.32	0.62 ± 0.18	0.67 ± 0.14	0.75
PonpokoAgent ⋆	2825.54 ± 3745.98	0.22 ± 0.23	0.96 ± 0.41	**0.73 ± 0.20**	**0.84 ± 0.09**	0.78
ParsCat2 ⋆	1992.60 ± 2674.90	0.13 ± 0.15	1.07 ± 0.30	0.66 ± 0.17	0.69 ± 0.14	0.86

Table 4.5: Performance comparison of *ANESIA* against other ANAC winning strategies averaged over all the 8 domains and two uncertain preference profiles in each domain. See supplement for the separate results for each domain. ANAC'19 agents (●) have uncertain user preferences, and no learning capabilities. ANAC'17 (⋆) and ANAC'18 (◇) agents can learn from experience and are given real user preferences. In blue are the best among ANESIA and ANAC'19 agents. In purple, the overall best.

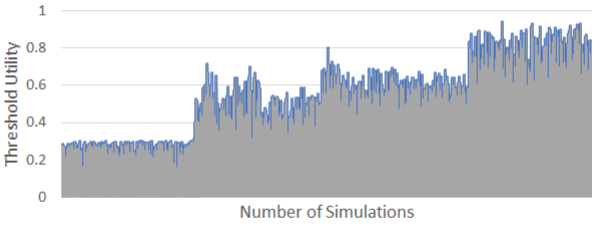

Figure 4.2: Increase in Dynamic Threshold Utility using DRL

During the experiments, we observed that *ANESIA* learns to increase the threshold utility in situations where the utility of an agreement is higher. Consequently, this makes an *ANESIA* agent push for deals where it maximizes its utility due to the higher threshold learned, which in turn results in ignoring low utility deals, and thus *ANESIA* agents have low success rate as compared to other agents. Figure 4.2 shows an example of such threshold utility increase over time in one of the domains (Grocery) against a set of unknown opponents (Kakesoba and SAGA).

We note that AgentHerb is the best in terms of P_{avg}, which is not surprising because this is one of the agents that know the true user model. This is clearly an unfair advantage over the agents, like *ANESIA*, that do not have this information. That said, ANESIA attains the best P_{avg} among ANAC'19 agents (unaware of the true user model) and the second best lowest P_{avg} among ANAC'17 and ANAC'18 agents (aware of the true user model). Even though they have an unfair advantage in knowing the true user model, we consider ANAC'17 and ANAC'18 agents since, like our approach, they enable learning from past negotiations.

To this end, we note that *ANESIA* uses prior negotiation data from AgentGG to pre-train the DRL-based utility threshold and adjust the selection of tactics from the templates. The effectiveness of our approach is demonstrated by the fact that *ANESIA* outperforms the same agents it was trained on (see Hypothesis C), but, crucially, does so also on domains and opponents unseen during training. We further stress that the obtained performance metrics are affected only in part by an adequate pre-training of the strategies: the quality of the estimated user and opponent models – derived without any prior training data from other agents – plays an important role too. The results in Tables 4.5 (A to C) evidence that our agent consistently outperforms its opponents in terms of individual and social efficiency, demonstrating that *ANESIA* can learn to adapt at run-time to different negotiation settings and against different unknown opponents.

Here, we conclude that the learned ANESIA strategy using 'strategy templates' can find the offers of higher individual and social welfare utility which are closer to the Pareto front even under the preference uncertainties, irrespective of the number of successful agreements.

4.6 Summary

This chapter introduced an agent negotiation model called *ANESIA*, which encapsulated different types of learning to support multi-issue negotiation under user preference uncertainty. *ANESIA* relied upon stochastic search based on the Firefly algorithm for user modelling and combined NSGA-II and TOPSIS for generating (near) Pareto-optimal bids. It further exploited strategy templates to learn the best combination of acceptance and bidding tactics at any negotiation time, and among its tactics, it used an adaptive target threshold utility learned using the DDPG algorithm. We have empirically evaluated the performance of *ANESIA* against the winning agent strategies of ANAC'17, '18 and '19 competitions in different settings, showing that our agent both outperforms opponents known at training time and can effectively transfer its knowledge to environments with previously unseen opponent agents and domains.

However, there are two main open issues with the work presented in this chapter. (a) The strategy template parameters of *ANESIA* were learned offline during training once, and then these fixed learned parameters were reused in all future negotiations. This results in a one-size-fits-all strategy that is not adaptive in settings that were not used during training. (b) In *ANESIA*, the estimated user model (computed once in the pre-negotiation phase – Phase-I) and the estimated opponent model (updated continuously during the negotiation phase – Phase-II), are sources of uncertainty which are not addressed during the generation of the (near) Pareto optimal bids. We address (a) in Chapter 5 (next chapter), and (b) in Chapter 6.

Chapter 5

Multiple-Issues Bilateral Negotiation Model - II

In this chapter, we use DRL throughout an actor-critic architecture *ANESIA*, discussed in the previous chapter, to estimate various tactic parameter values for strategy templates, in particular, (a) for a threshold utility, (b) when to accept an offer, and (c) how to generate a new bid. This contrasts with the previous chapter, in which we only estimate the threshold utility for those tactics in the template that require it using DRL. In Section 5.1, we present an argument of using DRL for learning the choice parameter values, which is followed by the discussion of proposed extension of *ANESIA* called *DLST-ANESIA* model in Section 5.2. Then, in Sections 5.3 and 5.4, we discuss materials and methods, and the experimental results respectively. Finally, we give the summary of this chapter in Section 5.5.

5.1 Motivation

Interpretable strategy templates, developed in the previous chapter, guide the use of a series of tactics whose optimal use can be learned during negotiation. The structure of such templates depends upon a number of learnable choice parameters, determining which acceptance and bidding tactic to employ at any particular time during negotiation. As these tactics represent hypotheses to be tested, defined by the agent developer, they can be explained to a user, and can in turn depend on learnable parameters. As an outcome, in the previous chapter, we formulated a strategy template for bid acceptance and generation so that an agent that uses it can make optimal decisions about the choice of tactics while negotiating in different domains [17].

The benefit of the above approach is that it can use heuristics for the components of the template and meta-heuristics or machine learning for evaluating the choice parameter values of these components. The problem with the previous chapter, however, is that the choice parameters of the components for the acceptance and bidding templates are learned once (during training) and used in all the different negotiation settings (during testing) [17]. This one-size-fits-all choice of tactics does not accumulate learning experience and may be unsuitable for unknown domains or unknown opponents. In other words, the mechanism for learning the choice parameter values used in the previous chapter [17] omits what is learned in a specific domain once the negotiation has finished, and therefore, cannot transfer the knowledge from one domain to new domains or unseen components.

To address the above limitation, we propose the idea of using DRL to estimate the choice parameter values of components in strategy templates. We name the proposed interpretable strategy templates as "Deep Learnable Strategy Templates (DLST)". Our contribution is that we study experimentally the ideas behind DLSTs so that agents that employ them to learn parameter values from and across negotiation experiences, hence being capable of transferring the knowledge from one domain to the other, or using the experience against one opponent on the other. This approach leads to adaptive and generalizable strategy templates. During all our experiments, we assume the same negotiation environment E as for *ANESIA* in Chapter 4.

5.2 The *DLST-ANESIA* Model

In this section, we present the proposed extended version of *ANESIA* called *DLST-ANESIA* as shown in Figure 5.1. When building a negotiation agent, we normally consider three phases: *pre-negotiation phase* (i.e., estimation of agent owner's preferences, preference elicitation), *negotiation phase* (i.e., offer generation, opponent modelling) and *post-negotiation phase* (i.e., assessing the optimality of offers) [77]. In this work, we are interested in the second phase, which involves a *Decide* component for choosing an optimal action a_t. As in previous chapter, we assume that our agent A_u is situated in an environment E (containing the opponent agent A_o) where, at any time t, A_u senses the current state S_t of E and represents it as a set of internal attributes, as shown in Figure 5.1; however this component was implicit in Figure 4.1 of the previous chapter.

To estimate the threshold utility in DLST-ANESIA, the set of state attributes include information derived from the sequence of previous bids offered by A_o (e.g., utility of the most recently received bid from the opponent ω_t^o, utility of the best opponent bid so far O_{best}, average utility of all the opponent bids O_{avg} and their variability O_{sd}) and information stored in A_u's internal state (e.g., number of bids B in the given partial order, d_D, u_{res}, Ω, and n), and the current negotiation time t. This internal state representation, denoted with s_t, is used by the agent (in acceptance and bidding strategies) to decide what action a_t to execute from the set of *Actions* based on the negotiation protocol P at time t. Action execution then changes the state of the environment to S_{t+1}. The state attributes for acceptance strategy involves the following attributes in addition to the above-mentioned ones: fixed target utility u, dynamic and learnable target utility $\bar{u}_t$, utility of the future bid ω (i.e., the bid to be proposed by our agent) $U(\omega)$, q quantile value which changes w.r.t time t, and utiltiy of the q^{th} best bid received by the agent A_o during the negotiation $Q_{\widehat{U}(\Omega_t^o)}(q)$. On the other hand, the state for bidding strategy involves the following set of state attributes: a bid generated by Time-dependent Boulware strategy

$b_{Boulware}$, Pareto-optimal bid PS, a recently received bid from the opponent with value of least important issue tweaked randomly $b_{opp}(\omega_t^o)$,and a random bid above average utility threshold $\mathcal{U}(\Omega_{\geq \bar{u}_t})$ (as discussed in the subsequent section), in addition to the state attributes used for estimating the dynamic threshold utility value.

Recall from Section 4.3, the action a_t is derived via two functions, f_a and f_b, for the acceptance and bidding strategies, respectively. The function f_a takes as inputs s_t, a *dynamic threshold utility* $\bar{u}_t$ (defined later in Section 5.3), the sequence of past opponent bids Ω_t^o, and outputs a discrete action a_t among *accept* or *reject*. When f_a returns *reject*, f_b computes what to bid next, with input s_t and $\bar{u}_t$, see ((4.1)–(4.2) in Section 4.3.1). This separation of acceptance and bidding strategies is not rare, see for instance [11]. Also, f_a and f_b consists of a set of tactics as defined in Section 4.3. We assume incomplete opponent preference information, therefore, *Decide* uses the estimated model $\widehat{U}_o$. In particular, $\widehat{U}_o$ is estimated at time t using information from Ω_t^o, see Section 4.4 for more details. Unlike the previous chapter, we employ DRL in acceptance strategy templates and bidding strategy templates in this chapter, in addition to threshold utility (represented by three green coloured boxes in Figure 5.1) in the *Decide* component. Each DRL component is based on actor-critic architecture [136] and has its own *Evaluate* and *Negotiation Experience* components.

Evaluate refers to a critic helping our agent learn the dynamic threshold utility $\bar{u}_t$, acceptance and bidding strategy template parameters, with the new experience collected during the negotiation against each opponent agent. More specifically, it is a function of random K $(K < N)$ experiences fetched from the agent's memory. Here, learning is *retrospective*, since it depends on the reward r_t obtained from E by performing action a_t at state s_t.

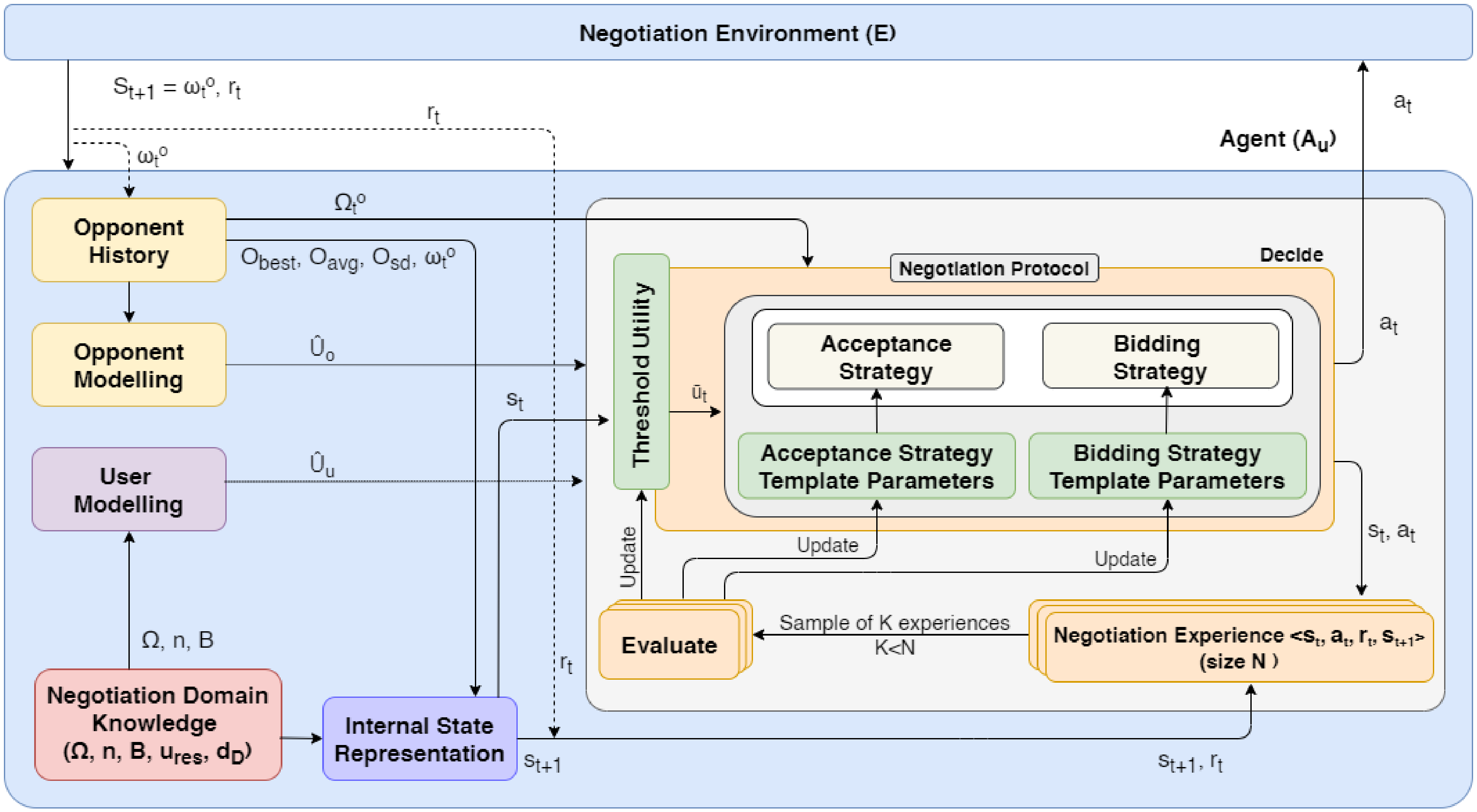

Figure 5.1: Interaction between the components of DLST-based agent negotiation model

The reward values for every critic that are used for estimating the threshold utility (i.e., $r_t^{\bar{u}_t}$ - same as that of (4.3) in ANESIA - see Chapter 4) as well as choice parameter values of acceptance (i.e., r_t^{bid}) and bidding strategy templates (i.e., r_t^{acc}) depend on the discounted user utility of the last bid received from the opponent ω_t^o, or of the bid accepted by either parties ω^{acc} and defined as (5.1), (5.2) and (5.3) respectively.

$$r_t^{\bar{u}_t} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement} \\ U_u(\omega_t^o, t), & \text{on received offer} \\ -1, & \text{otherwise.} \end{cases} \tag{5.1}$$

$$r_t^{bid} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement} \\ -1, & \text{otherwise.} \end{cases} \tag{5.2}$$

$$r_t^{acc} = \begin{cases} U_u(\omega^{acc}, t), & \text{on agreement and } U_o(\omega^{acc}, t) \leq U_u(\omega^{acc}, t) \\ U_u(\omega_t^o, t), & \text{on rejection and } U_o(\omega_t^o, t) \geq U_u(\omega_t^o, t) \\ -1, & \text{otherwise.} \end{cases} \tag{5.3}$$

$r_t^{\bar{u}_t}$ (5.1) and r_t^{bid} (5.2) are straight-forward. In (5.3), $U_o(\omega, t)$ is used as the reward value because reward is received from the environment E where the opponent agent resides. In other words, we assume that E has access to A_o's real preferences, i.e., U_o, but these preferences are not observable by our agent A_u. The first case of the r_t^{acc} deals with an agreed bid and returns a positive reward value, if the bid gives higher utility to our agent than the opponent. The second case deals with a rejected bid and returns a positive reward value, if the bid gives lower utility to our agent than the opponent. In all other cases, it returns a negative value.

Also, in (5.1), (5.2) and (5.3), $U_u(\omega, t)$ is the discounted reward of ω defined as defined in (4.4). In (4.4), d is a temporal discount factor to encourage the agent to negotiate without delay. If $d = 1$, the utility is considered undiscounted. We should not confuse d, which is typically unknown to the agent, with the discount factor used to compute the utility of an agreed bid (d_D).

Negotiation Experience stores historical information about N previous interactions of an agent with other agents. Experience elements are of the form $\langle s_t, a_t, r_t, s_{t+1} \rangle$, where s_t is the internal state representation of the negotiation environment E, a_t is the performed action, r_t is a scalar *reward* received from the environment and s_{t+1} is the new agent state after executing a_t.

Strategy templates We use the same notion of "strategy templates" as defined in Section 4.3.2 in Chapter 4. These are a general form of parametric strategies for acceptance and bidding. These strategies apply different tactics at different phases of the negotiation. The total number of phases n and the number of tactics n_i to choose from at each phase $i = 1, \ldots, n$ are the only parameters fixed in advance. For each phase i, the duration δ_i (i.e., $t_{i+1} = t_i + \delta_i$) and the particular choice of tactic are learnable parameters. The latter is encoded with choice parameters $c_{i,j}$, where $i = 1, \ldots, n$ and $j = 1, \ldots, n_i$, such that if $c_{i,j}$ is true then the (i, j)-th tactic is selected for phase i. Tactics can be parametric in turn, and depend on learnable parameters $\mathbf{p}_{i,j}$.

We consider the same set of admissible tactics as for *ANESIA* (see subsection 4.3.2). The key difference is that the new approach allows evolving the entire strategy (within the space of strategies entailed by the template) at every negotiation step, which makes it more adaptable and generalizable. Below, we give an example of a concrete acceptance strategy learned with our model. We use, as we will discuss in Section 5.4, a specific domain (*Party*) and we show how the strategy adapts in other negotiation domains (*Grocery* and *Outfit*) against the opponent strategy [17].

(a) Party Domain

$$\begin{aligned}
t \in [0.000, 0.0361) &\rightarrow U_u(\omega_t^o) \geq \max\Big(Q_{U_{\Omega_t^o}}(-0.20 \cdot t + 0.22), \bar{u}_t\Big)\\
t \in [0.0361, 1.000] &\rightarrow U_u(\omega_t^o) \geq \max\Big(u, Q_{U_{\Omega_t^o}}(-0.10 \cdot t + 0.64)\Big)
\end{aligned}$$

$$\downarrow$$

(b) Grocery Domain

$$\begin{aligned}
t \in [0.000, 0.1739) &\rightarrow U_u(\omega_t^o) \geq \max(u, \bar{u}_t)\\
t \in [0.1739, 0.2104) &\rightarrow U_u(\omega_t^o) \geq \max\Big(Q_{U_{\Omega_t^o}}(-0.10 \cdot t + 0.17), \bar{u}_t\Big)\\
t \in [0.2104, 1.000] &\rightarrow U_u(\omega_t^o) \geq \max(U_u(\omega_t), \bar{u}_t)
\end{aligned}$$

$$\downarrow$$

(c) Outfit Domain

$$\begin{aligned}
t \in [0.000, 0.0803) &\rightarrow U_u(\omega_t^o) \geq u\\
t \in [0.0803, 0.1829) &\rightarrow U_u(\omega_t^o) \geq \max\Big(\bar{u}_t, Q_{U_{\Omega_t^o}}(-0.33 \cdot t + 0.76)\Big)\\
t \in [0.1829, 0.2178) &\rightarrow U_u(\omega_t^o) \geq \max\Big(\bar{u}_t, Q_{U_{\Omega_t^o}}(-1.33 \cdot t + 0.99), U_u(\omega_t)\Big)\\
t \in [0.2178, 1.000) &\rightarrow U_u(\omega_t^o) \geq \bar{u}_t
\end{aligned}$$

We can observe that the duration learned in the left-hand side of the tactics is different for different domains, e.g., initially in the first domain ($Party$) the first rule triggers when $t \in [0.0, 0.0361)$, while in the second ($Grocery$) and third ($Outfit$) domains, the first rule triggers at $t \in [0.0, 0.1739)$ and $t \in [0.0, 0.0803)$ respectively. Similarly, the parameters on the right-hand side of the tactics rules, e.g., for the first domain ($Party$) during the very early phase of the negotiation, the strategy uses a quantile tactic as well as dynamic threshold utility. However, in the second domain ($Grocery$), the strategy now employs dynamic target threshold utility and the fixed threshold utility tactics, whereas, in the third domain ($Outfit$), it only employs the fixed utility tactic.

5.3 Setting Up *DLST-ANESIA* for Experiments

In our approach, we first use supervised learning (SL) to pre-train our agent using supervision examples collected from existing "teacher" negotiation strategies as inspired by [17, 15]. Such pre-trained strategy is then evolved via RL using experience and rewards collected while interacting with other agents in the negotiation environment. This combination of SL and RL approaches enhances the process of learning an optimal strategy. This is because applying RL alone from scratch would require a large amount of experience before reaching a reasonable strategy, which might hinder the online performance of our agent. On the other hand, starting from a pre-trained policy ensures quicker convergence (as demonstrated empirically in [17, 15]).

5.3.1 Data set Collection

In order to collect the data set for pre-training our agent via SL, we have used the *GENIUS* simulation environment [88]. In particular, in our experiments we generate supervision data using *ANESIA* agent proposed in previous chapter. by negotiating it against the winning strategies of ANAC-2019 competition, i.e., AgentGG, KakeSoba and SAGA. These strategies are readily available in GENIUS and requiring minimal changes to work for our negotiation settings. They also assume no user preference uncertainty in three different domains (Laptop, Holiday, and Party).

5.3.2 Strategy Representation

We represent both f_a (4.1) and f_b (4.2) using artificial neural networks (ANNs) [53], as these are powerful function approximators and benefit from extremely effective learning algorithms, unlike [17], which used the meta-heuristic optimization algorithm. We also use the same to predict the target threshold utility $\bar{u}_t$ as in [17]. Moreover, we keep the ANN configuration same as in Chapter 4. Furthermore, as in Section 4.5, we use Deep Deterministic Policy Gradient (DDPG) algorithm, which is an actor-critic RL approach, to generate a deterministic action selection policy for the negotiating agent [86].

In our experiments, for predicting the dynamic threshold utility, the actor function is a single-output regression ANN; on the other hand, for acceptance and bidding strategies, it is a multiple-output regression ANN. In particular, when predicting $\bar{u}_t$, act_t corresponds to $\bar{u}_t$; whereas, for acceptance and bidding strategy templates, act_t consists of a vector of multiple outputs $(\delta_i, (c_{i,j}, \mathbf{p}_{i,j})_{j=1,\ldots,n_i})_{i=1,\ldots,n}$ including the duration of each negotiation phase δ_i, Boolean choice parameters $c_{i,j}$ and a set of learnable parameters $\mathbf{p}_{i,j}$ for each tactic j that can be used in a negotiation phase i.

We consider a negotiation environment with uncertainty about the opponent's preferences. To derive an estimate of the opponent model $\widehat{U}_o$ during negotiation, we use the same distribution-based frequency model [142] as used in Chapter 4.

5.4 Experimental Setup and Results

All the experiments are performed using GENIUS [88]. The experiments are designed to prove the hypotheses defined below:

5.4.1 Experimental Hypotheses

- **Hypothesis A:** *DLST-ANESIA* outperforms *ANESIA* (proposed in previous chapter) and other "teacher" strategies in known negotiation settings in terms of individual and social efficiency.

- **Hypothesis B:** *DLST-ANESIA* outperforms unseen strategies and adapts to different negotiation settings in terms of individual and social efficiency.

5.4.2 Performance Metrics

As in Chapter 4, we measure the performance of each agent in terms of the same six widely-adopted metrics inspired by the ANAC competition:

- U_{ind}^{total}: The utility gained by an agent averaged over all the negotiations (↑);
- U_{ind}^{s}: The utility gained by an agent averaged over all the *successful* negotiations (↑);
- U_{soc}: The utility gained by both negotiating agents averaged over all successful negotiations (↑);
- P_{avg}: Average minimal distance of agreements from the Pareto Frontier (↓).
- $S_{\%}$: Proportion of successful negotiations (↑).

The first and second measures represent *individual efficiency* of an outcome, whereas the third and fourth correspond to the *social efficiency* of agreements.

5.4.3 Experimental Settings

Our proposed *DLST-ANESIA* model is evaluated against state-of-the-art strategies that participated in ANAC'17 and ANAC'18, which are designed by different research groups independently. Each agent has no information about another agent's strategies beforehand. Details of all these strategies are available in [7, 70]. We evaluate our approach on total of 11 negotiation domains which are different from each other in terms of size and opposition [8] to ensure good negotiation characteristics and to reduce any biases. The domain size refers to the number of issues, whereas opposition[1] refers to the minimum distance from all possible outcomes to the point representing complete satisfaction of both negotiation parties (1,1). For the experiments of Hypothesis B, we choose readily-available 3 small-sized, 2 medium-sized, and 3 large-sized domains. Out of these domains, 2 are with high, 3 with medium

[1] The value of opposition reflects the competitiveness between parties in the domain. Strong opposition means a gain of one party is at the loss of the other, whereas, weak opposition means that both parties either lose or gain simultaneously [8].

and 3 with low opposition (see [149] for more details).

For each configuration, each agent plays both roles in the negotiation (e.g., buyer and seller in Laptop domain) to compensate for any utility differences in the preference profiles. We call *user profile* the agent's role along with the user's preferences. Also, we set the u_{res} and d_D to their respective default values, whereas the deadline is set to 180s, normalized in $[0, 1]$ (known to both negotiating parties in advance). For NSGA-II during the Pareto-bid generation phase, we choose the population size of $2\% \times |\Omega|$, 2 generations and mutation count of 0.1. With these hyperparameters, on our machine[2] the run-time of NSGA-II never exceeded the given timeout of 10s for deciding an action at each turn, while being able to retrieve empirically good solutions.

5.4.4 Empirical Evaluation

In this section, we evaluate and discuss the two hypotheses introduced at the beginning of the section.

Hypothesis A: *DLST-ANESIA* outperforms *ANESIA* and other "teacher" strategies

We performed a total of 1200 negotiation sessions[3] to evaluate the performance of *DLST-ANESIA* agent against the four "teacher" strategies (ANESIA [17], AgentGG, KakeSoba and SAGA) in three domains (Laptop, Holiday, and Party - see Appendix A). These strategies were used to collect the dataset in the same domains for supervised training before the DRL process begins. Table 5.1 demonstrates the average results over all the domains and profiles for each agent. Clearly, *DLST-ANESIA* agent outperforms the "teacher" strategies in terms of individual efficiency, as well as social efficiency.

[2] CPU: 8 cores, 2.10GHz; RAM: 32 GB

[3] $n \times (n-1)/2 \times x \times y \times z = 1200$ where $n = 5$, number of agents in a tournament; $x = 2$, because agents play both sides; $y = 3$, number of domains; $z = 20$, because each tournament is repeated 20 times.

Agent	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
		Laptop Domain			
DLST-agent	**0.0 ± 0.0**	**1.71 ± 0.03**	**0.91 ± 0.02**	**0.91 ± 0.02**	**1.00**
ANESIA	**0.0 ± 0.0**	1.66 ± 0.20	0.86 ± 0.03	0.86 ± 0.03	**1.00**
KakeSoba	0.03 ± 0.12	1.48 ± 0.53	0.77 ± 0.20	0.82 ± 0.06	0.94
SAGA	0.01 ± 0.06	1.45 ± 0.48	0.89 ± 0.13	0.89 ± 0.10	0.99
AgentGG*	0.22 ± 0.35	1.14 ± 0.65	0.71 ± 0.38	**0.91 ± 0.09**	0.78
		Holiday Domain			
DLST-agent	**0.05 ± 0.11**	**1.74 ± 0.14**	**0.96 ± 0.14**	**0.96 ± 0.14**	**1.00**
ANESIA	0.06 ± 0.1	**1.74 ± 0.14**	0.85 ± 0.15	0.85 ± 0.15	**1.00**
KakeSoba	0.21 ± 0.35	1.53 ± 0.5	0.84 ± 0.27	0.92 ± 0.07	0.91
SAGA	0.19 ± 0.36	1.55 ± 0.5	0.70 ± 0.25	0.77 ± 0.12	0.91
AgentGG*	0.46 ± 0.58	1.16 ± 0.82	0.74 ± 0.45	**0.96 ± 0.03**	0.67
		Party Domain			
DLST-agent	**0.15 ± 0.38**	**1.53 ± 0.6**	**0.74 ± 0.31**	**0.77 ± 0.14**	**0.87**
ANESIA	0.37 ± 0.32	1.06 ± 0.5	0.52 ± 0.27	0.62 ± 0.14	0.83
KakeSoba	0.33 ± 0.32	1.11 ± 0.51	0.64 ± 0.3	0.75 ± 0.12	0.84
SAGA	**0.15 ± 0.16**	1.36 ± 0.26	0.61 ± 0.19	0.63 ± 0.16	**0.87**
AgentGG*	0.38 ± 0.42	0.92 ± 0.6	0.62 ± 0.4	**0.77 ± 0.12**	0.71

Table 5.1: Performance Comparison of *DLST-ANESIA* agent with “teacher” strategies for all the 3 domains (Laptop, Holiday, and Party - Readily available in GENIUS). Best Results are in **bold**. * means user preference uncertainty is considered.

Hypothesis B: Adaptive behaviour of *DLST-ANESIA* agent

We further evaluated the performance of a *DLST-ANESIA* agent against the opponent agents from ANAC’17 and ANAC’18 unseen during training and having capability of learning from previous negotiations. For this, we performed two experiments against ANAC’17 and ANAC’18 agents, each with a total of 29120 negotiation sessions[4]. Results in Table 5.2 are averaged over all domains, and demonstrate that a *DLST-ANESIA* agent learns to choose a suitable set of tactics to be used at run time and outperforms the other 8 strategies in terms of U_{ind}^{s} and U_{soc}. We also observed that *DLST-ANESIA* outperforms *ANESIA* in all the settings used for the purpose of evaluation as shown in Figures 5.2 – 5.5. This indicates that the *DLST-ANESIA* agent’s approach of dynamically adapting the parameters of acceptance and bidding strategies leads consistently improve the *ANESIA* approach of keeping these parameters fixed once the agent is deployed.

[4] $n \times (n-1)/2 \times x \times y \times z = 29120$ where $n = 14$; $x = 2$; $y = 8$; $z = 20$.

Agent	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
Comparison of DLST and ANESIA with ANAC 2017 Agent Strategies					
DLST-agent	**0.0 ± 0.0**	**1.17 ± 0.12**	**0.90 ± 0.0**	**0.93 ± 0.0**	**1.0**
ANESIA	**0.0 ± 0.0**	1.16 ± 0.12	0.70 ± 0.25	0.76 ± 0.26	0.89
PonpokoAgent	0.70 ± 0.49	0.44 ± 0.70	0.62 ± 0.19	**0.93 ± 0.04**	0.89
ShahAgent	0.54 ± 0.54	0.79 ± 0.79	0.57 ± 0.07	0.64 ± 0.04	0.75
Mamenchis	0.50 ± 0.05	0.80 ± 0.80	0.66 ± 0.16	0.82 ± 0.18	0.89
AgentKN	**0.0 ± 0.0**	**1.17 ± 0.0**	0.65 ± 0.05	0.65 ± 0.05	**1.0**
Rubick	1.08 ± 0.0	1.00 ± 0.0	0.50 ± 0.09	0.64 ± 0.04	0.76
ParsCat2	0.54 ± 0.54	0.80 ± 0.08	0.66 ± 0.16	0.82 ± 0.04	0.57
SimpleAgent	1.08 ± 0.0	0.90 ± 0.0	0.57 ± 0.14	0.57 ± 0.14	**1.0**
AgentF	1.18 ± 0.0	1.07 ± 0.06	0.51 ± 0.0	0.81 ± 0.0	0.89
TucAgent	0.08 ± 0.29	0.90 ± 0.03	0.65 ± 0.38	0.52 ± 0.16	0.69
MadAgent	0.67 ± 0.05	1.09 ± 0.17	0.57 ± 0.0	0.57 ± 0.0	**1.0**
GeneKing	1.08 ± 0.0	0.99 ± 0.14	0.75 ± 0.0	0.67 ± 0.24	0.63
Farma17	0.77 ± 0.49	0.44 ± 0.70	0.65 ± 0.19	**0.93 ± 0.04**	0.79
Comparison of DLST and ANESIA with ANAC 2018 Agent Strategies					
DLST-agent	**0.00 ± 0.08**	**1.54 ± 0.17**	**0.86 ± 0.07**	**0.87 ± 0.06**	**0.91**
ANESIA	**0.00 ± 0.09**	1.41 ± 0.16	0.74 ± 0.14	0.84 ± 0.14	0.78
AgentHerb	0.02 ± 0.05	0.79 ± 0.11	0.78 ± 0.02	0.78 ± 0.11	0.61
AgreeableAgent	0.05 ± 0.11	1.12 ± 0.23	0.53 ± 0.10	0.56 ± 0.05	0.54
Sontag	0.03 ± 0.07	0.73 ± 0.18	0.78 ± 0.08	0.79 ± 0.07	0.59
Agent33	0.04 ± 0.07	0.74 ± 0.18	0.68 ± 0.09	0.78 ± 0.09	0.79
AngentNP1	0.04 ± 0.06	0.73 ± 0.16	0.65 ± 0.10	0.65 ± 0.1	0.69
FullAgent	0.02 ± 0.04	0.67 ± 0.12	0.69 ± 0.05	0.77 ± 0.12	0.61
ATeamAgent	0.09 ± 0.06	0.58 ± 0.13	0.75 ± 0.10	0.75 ± 0.08	0.75
ConDAgent	0.06 ± 0.09	1.16 ± 0.20	0.68 ± 0.11	0.65 ± 0.11	0.56
GroupY	0.03 ± 0.06	0.66 ± 0.15	0.53 ± 0.07	0.54 ± 0.06	0.58
Yeela	0.04 ± 0.06	0.68 ± 0.14	0.73 ± 0.08	0.73 ± 0.07	0.66
Libra	0.10 ± 0.09	0.54 ± 0.19	0.71 ± 0.08	0.56 ± 0.04	0.77
ExpRubick	**0.00 ± 0.02**	1.10 ± 0.18	0.78 ± 0.08	0.80 ± 0.12	**0.91**

Table 5.2: Performance Comparison of *DLST-ANESIA* agent with existing strategies averaged over all the 8 domains (Airport Site, Camera, Energy, Fitness, Flight, Grocery, Itex-Cypress, Outfit - All are readily available in GENIUS). Best Results are in **bold**.

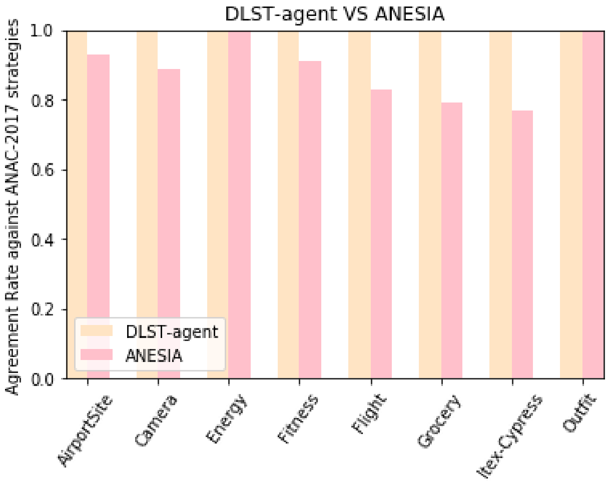

Figure 5.2: Comparison of *DLST-ANESIA* VS ANESIA agents in terms of Agreement rate $S_{\%}(\uparrow)$

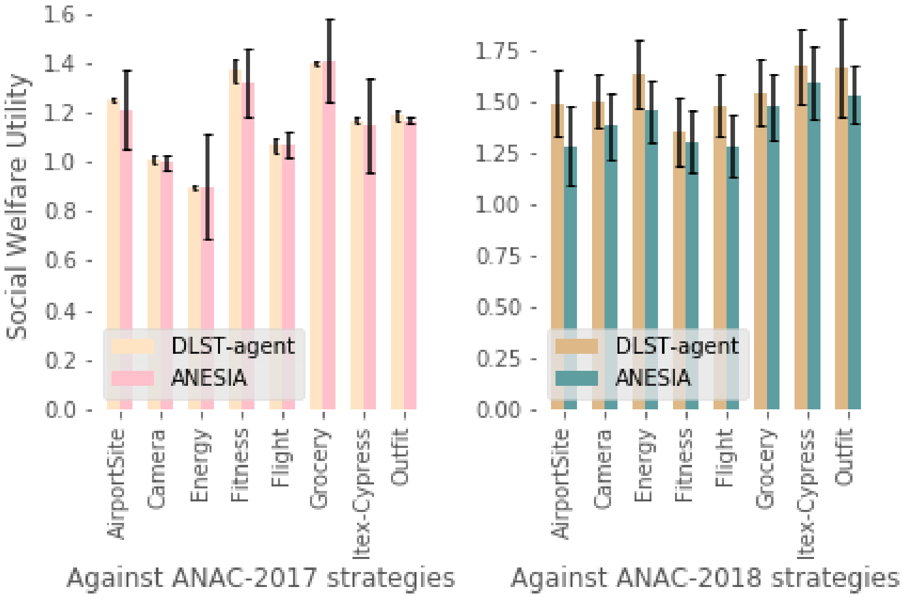

Figure 5.3: Comparison of *DLST-ANESIA* VS ANESIA agents in terms of Social welfare utility $U_{soc}(\uparrow)$

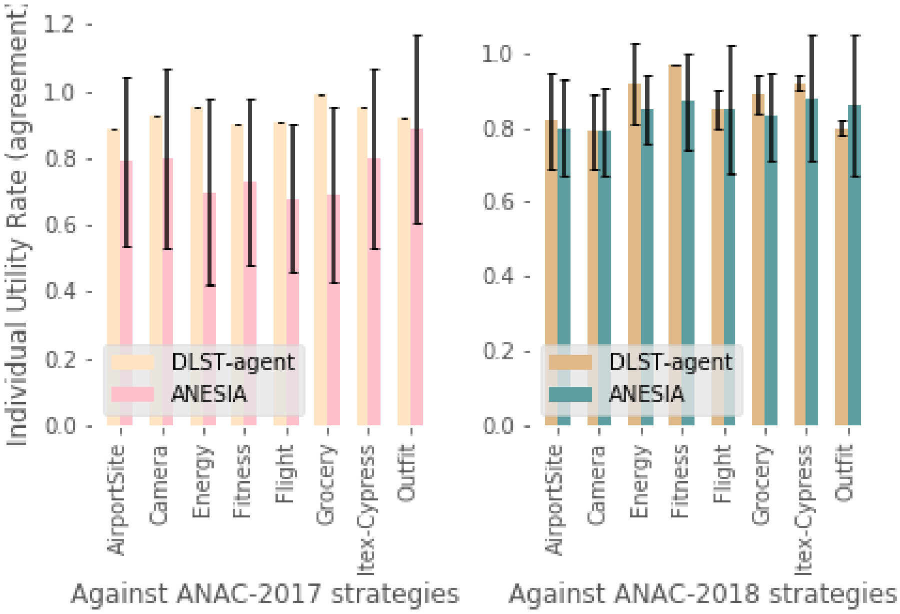

Figure 5.4: Comparison of *DLST-ANESIA* VS ANESIA agents in terms of individual utility rate over successful negotiations $U^s_{ind}(\uparrow)$

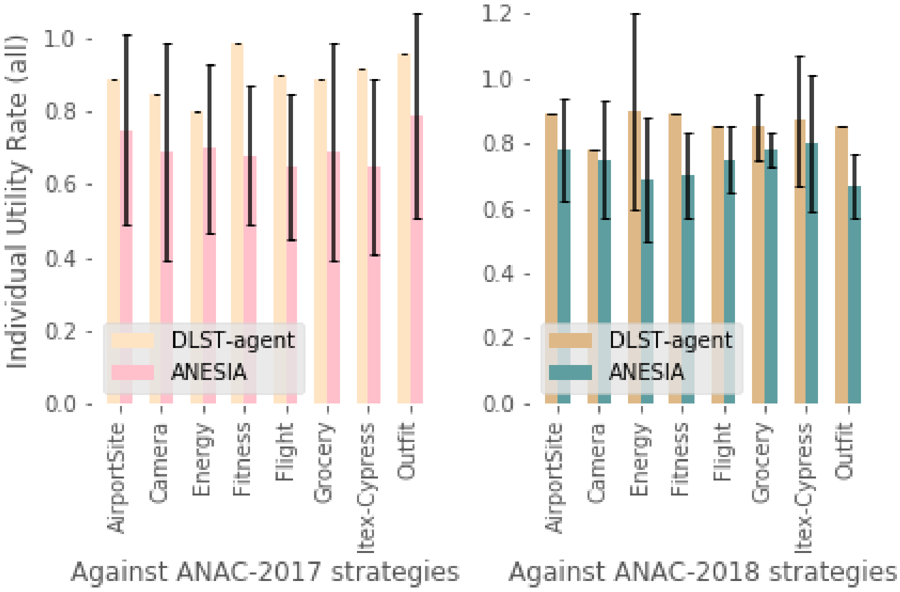

Figure 5.5: Comparison of *DLST-ANESIA* VS ANESIA agents in terms of individual utility rate over all negotiations $U^{total}_{ind}(\uparrow)$

Here, we conclude that the learned ANESIA strategy using Deep learning-based strategy templates can find the offers of higher individual and social welfare utility, while targetting the higher number of successful agreements.

5.5 Summary

In this chapter, we described *DLST-ANESIA*, an actor-critic architecture based DRL to support negotiation in domains with multiple issues. In particular, we exploited "interpretable" strategy templates used in the state-of-the-art to learn the best combination of acceptance and bidding tactics at any negotiation time. Among the *DLST-ANESIA* tactics, we used an adaptive threshold utility, all learned using the DDPG algorithm which derives an initial neural network strategy via supervised learning. We also showed the empirical performance evaluation of our *DLST-ANESIA* model against *ANESIA* and other "teacher strategies". We also performed comparison with the agent strategies of ANAC'17 and ANAC'18 competitions (since the tournament allowed learning from previous negotiations) in different settings. We observed that *DLST-ANESIA* agent outperforms opponent agents which are known at training time and can effectively transfer its knowledge to environments with previously unseen opponent agents and domains.

Chapter 6

Dealing with Uncertainty in Preferences

In this chapter, we are interested in the uncertainty that arises in the first two phases of automated negotiation (namely, *pre-negotiation phase* and *negotiation phase* - see Section 2.1.4). In *pre-negotiation phase*, the uncertainty is due to the estimation of the user's preferences/utility function, while in *negotiation phase*, the uncertainty arises due to the estimation of opponent's preferences/utility function. These uncertainties present a critical and sensitive obstacle because it may influence the bid search process and consequently hamper the identification of efficient solutions. In this context, we start off with the argument of extending *ANESIA* to *fuzzy-ANESIA* in Section 6.1. Then, in Section 6.2, we propose a two-phase process of generating the (near) Pareto-optimal bids under incomplete preference information combining MOO and uncertainty modelling with fuzzy techniques. Then, in Section 6.3, we empirically evaluate the proposed method in a range of negotiation settings and scenarios. Finally, Section 6.4 summarizes the chapter.

6.1 Motivation

Our proposed model known as *fuzzy-ANESIA* is an extension of the *ANESIA* model discussed in Chapter 4. In particular, *ANESIA* assumed incomplete information

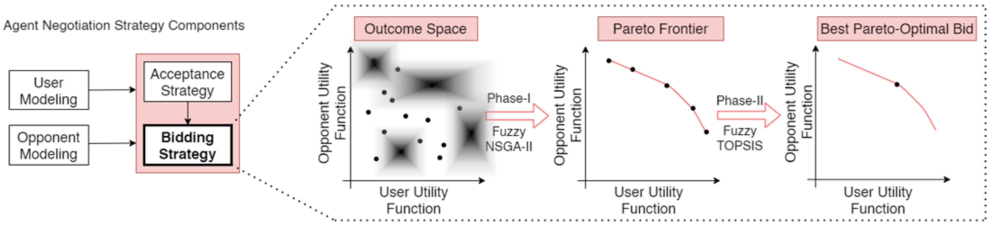

Figure 6.1: Two-phase process of generating a Pareto-optimal solution during bid generation phase

about the preferences of the user and opponent agents, and therefore estimated them before taking any action. More specifically, *ANESIA* didn't take into account the uncertainty that arose from the approximation of real user's and opponent's preferences while generating Pareto-optimal bids during the negotiation. As a result, many of the envisaged agreements were not very close to the Pareto frontier. So Chapter 4 opened an opportunity for addressing the issues with the estimated utility models in the bidding phase of the negotiation. More precisely, we address the uncertainties in the objectives of a MOO problem by means of triangular fuzzy numbers (see Section 2.2.6), and call the new approach *fuzzy-ANESIA*.

6.2 The *fuzzy-ANESIA* Model

In this section, we present the proposed two-phase solution of generating (near) Pareto-optimal bids as shown in Figure 6.1 and provide the background on the fuzzy methodologies used in this work.

In *fuzzy-ANESIA* model, the user modelling (i.e., estimation of user utility function) is done before the negotiation begins with the help of Cuckoo Search Optimization (CSO) [153], whereas opponent modelling is done with the help of distribution-based frequency model [142] (see also 4.4) during the negotiation. CSO [153] is a meta-heuristic inspired by the brood parasitism of cuckoo birds. In this metaphor, a cuckoo is an agent in search of its best user model $\widehat{U}$ (or nest or solution). A set of candidate solutions (user models) is evolved, where at each iteration the worst-performing p solutions are abandoned and replaced with new solutions generated by Lévy flights, that is, a kind of random walk combining long and short random movements. In our case, the fitness of a candidate solution $\widehat{U}'$ is defined as the Spearman's rank correlation coefficient ρ between the estimated ranking of $\widehat{U}'$ and the true, but partial, bid ranking $\preceq$. The coefficient $\rho \in [-1, 1]$ is indeed a measure of the similarity between two rankings, which assigns a value of 1 for identical rankings, and -1 for opposed rankings.

6.2.1 Phase I

In Phase-I, we address the effects of uncertainty propagation in the multi-objective setting where uncertainty is assumed to occur in the objective functions i.e., user and opponent utility functions because of lack of information. We use an extended-NSGA-II [19], which replaces the classic Pareto dominance with the fuzzy Pareto dominance. Let Y and Y' be two triangular fuzzy solutions. Y strong dominates Y', if either y_i total dominates or partial dominates y_i' in one objective and weak[1] dominates it in another [19]. This modification allows ensuring the fitness assignment ranking in a fuzzy setting. Afterwards, a crowding-comparison procedure is applied based on a Crowding Distance (CD) that discriminates the solutions having the same rank level. Formally, the CD of a solution is the sum of its individual objectives' distances, that in turn are the differences between the solution and its closest neighbours as shown in (6.1).

$$CD(i) = \sum_{i=1 \ldots n} (f_i(i+1) - f_i(i-1))/(f_i^{max} - f_i^{min}) \; s.t. \; i \in F \tag{6.1}$$

In (6.1), n is the number of objectives, $f_i(i+1)$ and $f_i(i-1)$ are the neighbour objective values of the i^{th} objective, f_i^{max} and f_i^{min} are the maximum and minimum objective values respectively in the population, and F is the i^{th} front to which solutions are associated. Since our objective functions are TFN vectors (Recall from Section 2.2.6, TFN represents a triplet of values), the distance measure must

[1] y_i partially weak dominates y_i' iff there is fuzzy overlapping or fuzzy inclusion.

be adapted to fuzziness. Thus, these objectives are approximated by computing their expected values before applying CD. The expected value E of a given TFN $y_i = [\underline{y_i}, \widehat{y}, \overline{y_i}]$ (see (**??**)) is calculated as shown in (6.2):

$$E(y_i) = (\underline{y_i} + 2 \times \widehat{y_i} + \overline{y_i})/4 \tag{6.2}$$

To reflect the uncertainty in fuzzy objective values caused by the possibility of multiple utility functions as solutions, we collected $K = 100$ best user models derived by CSO. Since, the fitness value for each objective function is a TFN, $\widehat{a}$ is the average utility of the K models for that bid, and the upper and lower bounds are 5^{th} and 95^{th} percentile of the k utility models respectively. Given this distribution of utility values, we derive the principles' triangular sets as triangular approximation of the empirical normal approximation of the distribution. That is, let $\hat{\sigma}$ be the observed standard deviation, $\hat{\mu}$ be the observed mean, then the fuzzy-set could be: $[l, m, u]$ where $m = \hat{\mu}$, $l = \hat{\mu} - k\hat{\sigma}$ and $u = \hat{\mu} + k\hat{\sigma}$ for some $k = 1, 2, 3$ (we choose $k = 2$). The membership U of each element is given by the corresponding Gaussian probability density function value (see (6.3)).

$$U(x) = \frac{1}{\sqrt{2\pi}\hat{\sigma}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\hat{\sigma}}\right)^2} \tag{6.3}$$

Substituting the values of l, m and u in (6.3), we obtained $U(m) = 1/(\hat{\sigma}\sqrt{2\pi})$, $U(l) = U(u) = e^{-k^2/2}/(\hat{\sigma}\sqrt{2\pi})$. For the fuzzy opponent objective function, $\widehat{a}$ is the current opponent model at any time t, the upper and lower bounds are opponent models obtained at time $t \in [0.4, 0.6]$ and $t \leq 0.2$ respectively. Note that the percentiles are calculated to obtain the objective function (user model and opponent model) values, which are triplets. The normal approximation is used as a fitness function of NSGA-II, which are also triplet values.

The fuzzy NSGA-II generates a Pareto-frontier which has numerous (P) optimal solutions $(P \in \Omega)$ preferred according to the decision-making requirements. Hence, decision-making approaches are essential to pick an individual solution from the Pareto Frontier.

6.2.2 Phase-II

We employ a fuzzy Multi-Criteria Decision-Making (MCDM) method called fuzzy TOPSIS [32] to pick the best optimal solution from Pareto Frontier. In our model, we have only two criteria ($m = 2$) or objectives: maximizing user utility and maximizing opponent utility (since our focus is more on "win-win" situations), based on which Ω solutions/bids/alternatives will be ordered. Our agent implements fuzzy TOPSIS with the help of vertex method to calculate the distance between two triangular fuzzy numbers $y = (y_1, \ldots, y_m)$ and $y' = (y'_1, \ldots, y'_m)$ as follows:

$$\sqrt{1/3[(y_1 - y'_1)^2 + \ldots + (y_m - y'_m)^2]} \tag{6.4}$$

The procedure of fuzzy TOPSIS is defined as follows:

- A fuzzy decision matrix $M = n \times m$ consisting of n alternatives and m criteria is created. Here, $n = |P|$, $m = 2$, and $m_1 = \widehat{U}(\omega_i)$ and $m_2 = \widehat{U}(\omega_i^o)$.
- The next step is normalizing the fuzzy decision matrix M using (6.5), where $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$ and x_{ij} is a value assigned to the i^{th} solution w.r.t j^{th} criteria.

$$\begin{gathered} \tilde{r}_{ij} = \left(\frac{\underline{y_1}}{c_j^*}, \frac{\widehat{y_1}}{c_j^*}, \frac{\overline{y_1}}{c_j^*} \right), and \\ c_j^* = max_i\{c_{ij}\} \end{gathered} \tag{6.5}$$

- In the subsequent step, we create a weighted normalized decision matrix W using where x_{ij} is replaced with v_{ij} and $v_{ij} = w_j \cdot r_{ij}$. In our experiments, we use the same weights which were learned in the existing $ANESIA$ model. These weights scale with negotiation time t. So, $w_1 = a \cdot t + b$ and $w_2 = 1 - (a \cdot t + b)$. Here, $a = -0.75$ and $b = 0.6$.

- Once the weighted normalized matrix is ready, the distance of each alternative from fuzzy ideal positive and negative solutions is computed.

- Finally, the ranks are ordered from high to bottom based on the relative closeness of each alternative to the ideal solutions.

A bid solution/bid/alternative with top rank is chosen by our agent to propose to the opponent agent during this time period.

6.3 Experimental Results and Discussions

All the experiments have been performed using the popular GENIUS negotiation platform [88] simulating our assumed complex negotiation environment.

6.3.1 Experimental Hypotheses

These experiments are designed to prove the following two hypotheses:

- **Hypothesis A:** Fuzzy versions of existing negotiation strategies outperform their non-fuzzy variants in terms of U_{ind}, $Dist_p$ and U_{soc}.

- **Hypothesis B:** *fuzzy-ANESIA* outperforms the *ANESIA* agent and other winning agents from ANAC'19 competition in terms of U_{ind}, $Dist_p$ and U_{soc}.

6.3.2 Performance Metrics

We consider the same (widely adopted) metrics [10] inspired by the GENIUS simulation platform:

- R_{avg}: Average number of rounds over all successful negotiations (Ideal value: Low(1))

- $Dist_p$: Average distance to the Pareto Curve[2] (the nearest bid on the frontier) (Ideal value: Low (0))

[2] Pareto frontier is obtained assuming complete preference information of both the negotiating parties.

- U_{ind}: Average utility gained by an agent on successful negotiations (Ideal value: High (1.0)

- U_{soc}: Average utility gained by both negotiating agents on successful negotiations (Ideal value: High(2.0)

- $S_{\%}$: Proportion of successful negotiations (Ideal value: High (100%))

6.3.3 Experimental Settings

We assume that prior to designing an agent's negotiation strategy: (a) each agent has no knowledge of the preferences and negotiating characteristics of its opponent; (b) the negotiation time is limited and there is a specific deadline (known to both negotiating parties in advance) for its termination (here, it is $60s$ normalised in $[0, 1]$), therefore the agents must consider the risk of rejecting their offer from the opponent with regard to the limited time; (c) the utility of offers might decrease over time (in negotiation scenarios with discount factor; we use the default value in GENIUS) [17]), thus, timely decision on rejecting or accepting an offer and making acceptable offers are of high importance for negotiators.

We evaluate our approach on the same benchmark domains used in [17], i.e., Laptop ($|\Omega| = 27$), Holiday ($|\Omega| = 1024$) and Party ($|\Omega| = 3072$) (see Appendix A). For each configuration, each agent gets the chance to play both sides of the negotiation (e.g., buyer and seller in Laptop domain). We call *user profile* the specific agent's role along with its associated preferences.

6.3.4 Empirical Evaluation

Hypothesis A: Fuzzy versions outperform their Non-fuzzy variants

We performed the analysis of fuzzy hybrid approach of generating (near) Pareto-optimal bids by combining it with different negotiation strategies dealing with user's and opponent's preference uncertainties. We used 12 combinations of negotiations strategies involving 2 population-based user modelling approaches (Cuckoo Search CS and Genetic Algorithm GA), whereas 3 opponent-modelling approaches[3] (Bayesian model, Smith Frequency Model, and a Simple/Uniform model) with and without the component of Fuzzy Hybrid approaches for the total of 6600 simulations, each in 2 different domains (Party and Holiday) with 3 different user profiles ($B = 0.2 \times |\Omega|, 0.4 \times |\Omega|, 0.6 \times |\Omega|$). Tables 6.1 and 6.2 show that the negotiation strategies involving fuzzy component (starting with $'f'-$) outperform their non-fuzzy variants, mainly in terms of U_{ind}, $Dist_p$ and U_{soc} leading to "win-win" situations.

Hypothesis B: *fuzzy-ANESIA* outperforms ANESIA and other winning agents from ANAC'19

We also tested *fuzzy-ANESIA* in a GENIUS tournament setting against *ANESIA* and winning agents from the ANAC'19 competition[4], for a total of 1200 sessions in 3 different domains, where each agent negotiates with every other agent. Table 6.3 compares their performance in terms of U_{ind}, R_{avg}, and $S_{\%}$. Figure 6.2 shows the increase in U_{soc} of *fuzzy-ANESIA* agent, whereas Figure 6.3 shows the decrease in $Dist_{pareto}$ w.r.t original *ANESIA* agent. Results demonstrate that our proposed fuzzy approach of generating (near) Pareto optimal bids has significantly impacted the performance of *ANESIA* agent. In this experiment, we chose two different user profiles and two different preference uncertainties ($|B| \in \{10, 20\}$).

In addition, the low successful negotiation rate in Tables 6.1 and 6.2 with high U_{ind} and high U_{soc} (and in Figure 6.2) indicates the non-greedy behaviour of *fuzzy-ANESIA* agent, which is often seen in the agents belonging to the same institution, when they want to achieve the maximum mutual benefit instead of reaching an agreement which can be less beneficial to one of them.

[3] Available in GENIUS.

[4] *SAGA* (Genetic algorithm), *KakeSoba* (Tabu Search), and *AgentGG* (Statistical frequency modelling)

Table 6.1: Performance comparison of fuzzy VS non-fuzzy negotiation strategies for Holiday Domain ($B = 0.2\Omega, B = 0.4\Omega, B = 0.6\Omega$) (Best results in **bold** are chosen by pairwise comparison of fuzzy and non-fuzzy approaches)

Metric	f-GA-Smith	f-GA-Uniform	f-GA-Bayesian	f-CS-Smith	f-CS-Uniform	f-CS-Bayesian
$R_{avg}(\downarrow)$	(**29.18**, **24.69**, **25.34**)	(479.78, 523.15, 553.97)	(**678.20**, 676.58, 437.88)	(**166.16**, **94.28**, **61.96**)	(**111.63**, **82.84**, **79.19**)	(**916.12**, 876.43, **607.14**)
$Dist_p(\downarrow)$	(**0.16**, **0.16**, **0.10**)	(0.18, **0.16**, **0.12**)	(**0.14**, **0.14**, **0.10**)	(**0.14**, **0.14**, **0.10**)	(**0.15**, **0.13**, **0.08**)	(**0.16**, 0.16, **0.11**)
$U_{soc}(\uparrow)$	(**1.59**, **1.59**, **1.67**)	(**1.58**, **1.60**, 1.64)	(**1.54**, **1.58**, **1.64**)	(**1.61**, **1.62**, **1.68**)	(**1.60**, **1.62**, **1.70**)	(**1.59**, **1.60**, **1.66**)
$U_{ind}(\uparrow)$	(**0.78**, **0.81**,0.84)	(0.76, **0.78**, **0.87**)	(**0.76**, **0.79**, **0.87**)	(**0.83**, **0.82**, **0.86**)	(**0.80**, **0.81**, **0.85**)	(0.78, 0.80, **0.85**)
Metric	GA-Smith	GA-Uniform	GA-Bayesian	CS-Smith	CS-Uniform	CS-Bayesian
$R_{avg}(\downarrow)$	(489.46, 327.08, 378.22)	(**436.53**, **344.19**, **285.96**)	(771.89, **609.39**, **400.27**)	(610.97, 530.02, 285.61)	(692.11, 888.04, 521.09)	(946.87, **834.125**, 673.05)
$Dist_p(\downarrow)$	(0.18, 0.17, 0.12)	(**0.17**,0.17,0.11)	(0.18, 0.17, 0.14)	(0.16, 0.16,0.13)	(0.15, 0.14, 0.11)	(**0.16**, **0.15**, 0.12)
$U_{soc}(\uparrow)$	(1.56, 1.58, 1.65)	(1.56, 1.57, **1.66**))	(1.56, **1.58**, 1.63)	(1.59, 1.60, 1.63)	(**1.60**, 1.61, 1.66)	(**1.59**, **1.60**, 1.64)
$U_{ind}(\uparrow)$	(0.77, 0.78, **0.86**)	(**0.78**, **0.78**, 0.81)	(0.68, **0.79**, 0.80)	(0.80, 0.81, 0.82)	(**0.80**, **0.81**, **0.85**)	(**0.81**, **0.81**, 0.84)

Table 6.2: Performance comparison of fuzzy VS non-fuzzy negotiation strategies for Party domain ($B = 0.2\Omega, B = 0.4\Omega, B = 0.6\Omega$) (Best results in **bold** are chosen by pairwise comparison of fuzzy and non-fuzzy approaches)

Metric	f-GA-Smith	f-GA-Uniform	f-GA-Bayesian	f-CS-Smith	f-CS-Uniform	f-CS-Bayesian
$R_{avg}(\downarrow)$	(**2.65**, **2.71**,**61.11**)	(926.38, 2090.4, 1887.62)	(**1463.56**, 1709.94, 1887.42)	(**2.86**, **2.71**, **2.50**)	(**2.72**, **2.64**, **3.0**)	(2408.59, 1743.44, 4999.87)
$Dist_p(\downarrow)$	(**0.12**, **0.13**, **0.07**)	(**0.12**, **0.11**, 0.14)	(**0.17**, **0.16**, **0.14**)	(**0.11**, 0.14, **0.08**)	(0.14, **0.08**, **0.06**)	(**0.16**, **0.11**, **0.08**)
$U_{soc}(\uparrow)$	(**1.45**, **1.46**, **1.51**)	(**1.44**, **1.34**, **1.39**)	(**1.36**, **1.37**, **1.5**)	(**1.43**, **1.45**, **1.55**)	(**1.43**, **1.52**, **1.59**)	(**1.39**, 1.39, 1.42)
$U_{ind}(\uparrow)$	(**0.67**, **0.72**, **0.78**)	(**0.74**, 0.66, **0.69**)	(**0.65**, **0.66**, **0.69**)	(0.74, **0.79**, **0.81**)	(**0.77**, **0.80**, **0.94**)	(**0.67**, **0.71**, **0.85**)
Metric	GA-Smith	GA-Uniform	GA-Bayesian	CS-Smith	CS-Uniform	CS-Bayesian
$R_{avg}(\downarrow)$	(687.0, 782.48, 839.76)	(**791.81**, **798.13**, **1107.12**)	(1670.60, **1538.79**, **1766.26**)	(955.28, 1308.46, 2827.50)	(798.61, 1069.97, 2062.0)	(**1833.04**, **1626.81**, **1809.86**)
$Dist_p(\downarrow)$	(0.13,0.13, 0.10)	(0.14, 0.12, **0.12**)	(0.18,0.17,0.14)	(0.14, **0.13**,0.15)	(**0.13**, 0.13, 0.11)	(0.17,0.16,0.18)
$U_{soc}(\uparrow)$	(**1.45**, 1.45, 1.44))	(1.42, 1.33, 1.33))	(1.33, 1.30, 1.32)	(**1.43**, 1.43,1.45)	(1.42, 1.42, 1.49)	(1.35, **1.4**, **1.5**)
$U_{ind}(\uparrow)$	(0.61, 0.70, 0.75)	(0.70, **0.69**, 0.66)	(0.61,0.63,0.63)	(**0.75**, 0.73, 0.70)	(0.75, 0.74, 0.70)	(0.50,0.69,0.75)

Table 6.3: Performance comparison of *fuzzy-ANESIA* (with Strategy Template) VS Winning agents from ANAC'19 (Best results are in **bold**)

Metric	fuzzy-ANESIA	AgentGG	KakeSoba	SAGA
		Laptop domain ($B = 10, B = 20$)		
$U_{ind}(\uparrow)$	**(0.98 ± 0.0044, 0.99 ± 0.0124)**	(0.90 ± 0.0072, 0.88 ± 0.0128)	(0.88 ± 0.0096, 0.93 ± 0.0008)	(0.83 ± 0.0078, 0.79 ± 0.0039)
$R_{avg}(\downarrow)$	**(66.46 ± 20.33, 64.85 ± 23.00)**	(2575.71 ± 5086.512, 5089.774 ± 8463.00)	(2081.211 ± 2965.04, 4991.53 ± 3578.63)	(263.99 ± 261.81, 885.83 ± 759.40)
$S_{\%}(\uparrow)$	(42.12 , 52.5)	(70.5 , 58.17)	(**79.67** ,61.33)	(77.83 , **74.83**)
		Holiday domain ($B = 10, B = 20$)		
$U_{ind}(\uparrow)$	(**0.92 ± 0.007, 0.93 ± 0.055)**	(0.87 ± 0.0036, 0.89 ± 0.021)	(0.85 ± 0.0136, 0.85 ± 0.044)	(0.76 ± 0.0291, 0.74 ± 0.0087)
$R_{avg}(\downarrow)$	**(158.73 ± 660.71 , 135.22 ± 732.85)**	(848.37 ± 313.98, 441.29 ± 546.79)	(677.70 ± 540.84, 319.63 ± 724.39)	(510.91 ± 3035.42, 466.94 ± 284.32)
$S_{\%}(\uparrow)$	(66.83, 62.17)	(68.67, 77.17)	(74.00, **82.00**)	(**74.83**, 73.00)
		Party domain ($B = 10, B = 20$)		
$U_{ind}(\uparrow)$	**(0.92 ± 0.025, 0.90 ± 0.025)**	(0.76 ± 0.044, 0.75 ± 0.039)	(0.77 ± 0.11, 0.89 ± 0.0039)	(0.55 ± 0.042, 0.54 ± 0.0471)
$R_{avg}(\downarrow)$	**(100.00 ± 673.16, 123.06 ± 523.17)**	(644.32 ± 933.16, 735.55 ± 886.89)	(669.011 ± 932.39, 781.80 ± 562.44)	(428.18 ± 972.89, 395.22 ± 835.17)
$S_{\%}(\uparrow)$	(24.83, 25.00)	(60.33, 61.83)	(60.33, 59.83)	**(70.17, 71.83)**

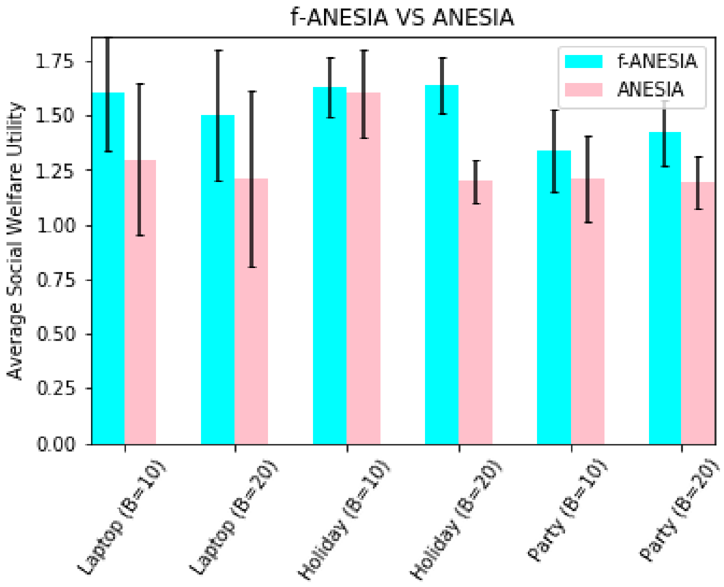

Figure 6.2: Average Social Welfare Utility (↑): fuzzy-ANESIA Vs ANESIA in 3 different domains with 2 user profiles $|B| = \{10, 20\}$

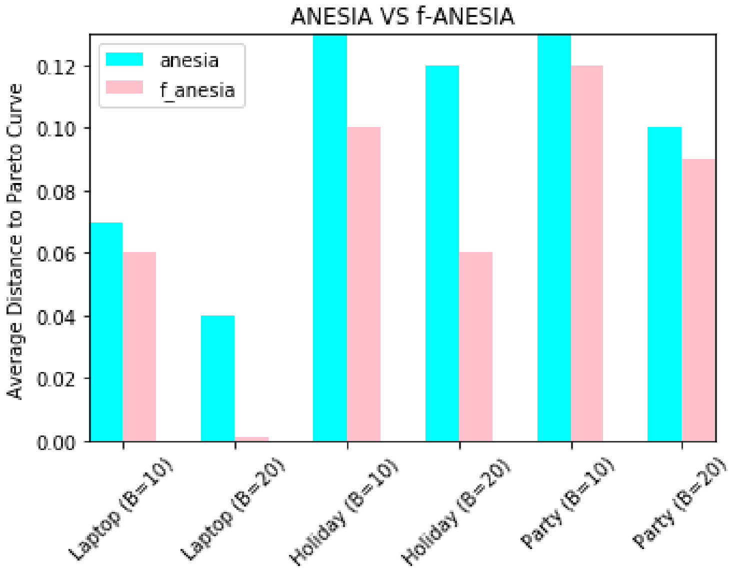

Figure 6.3: Average Distance to Pareto Curve (↓): fuzzy-ANESIA VS ANESIA in 3 different domains with 2 user profiles $|B| = \{10, 20\}$

Here, we conclude that the learned strategy of fuzzy-ANESIA finds the offers of higher individual and social-welfare utility which are closer to the Pareto front. In other words, it is becoming picky and non-greedy as our other trained strategies do in the previous chapters, even under the preference uncertainties of both the negotiating participants.

6.4 Summary

In Chapter 4, *ANESIA* relied on the meta-heuristic optimization for estimating the user preferences in the *pre-negotiation phase*, and the combination of MOO and MCDM to generate (near) Pareto-optimal bids in the *negotiation phase*. However, *ANESIA* did not account for how uncertainty propagated from the first to the second phase before generating (near) Pareto-optimal bids. To address this uncertainty, in this chapter, we presented *fuzzy-ANESIA* agent, an extension of *ANESIA*, which explored the use of amalgamation of fuzzy MOO and fuzzy MCDM methods. To evaluate the performance of our fuzzy approach, we first compared fuzzy and non-fuzzy variants of six different negotiation strategies in different negotiation domains. Then, we compared the performance of *fuzzy-ANESIA* against *ANESIA* and the winning agents of ANAC'19 competition, where all the agents are dealing with their owner's preference uncertainties, and span a wide range of strategies and techniques.

Chapter 7

Conclusions and Future work

In this chapter, we provide an overall summary of the research presented, the main findings and its contribution to the area of automated bilateral negotiation. We also present a discussion of future research directions. We start off with the summary in Section 7.1 which is followed by the main findings of the research in Section 7.2. The research contributions are presented in Section 7.3. Lastly, Section 7.4 presents some considerations for future work.

7.1 Thesis Summary

In this thesis, we presented a novel Deep Reinforcement Learning (DRL) model based on the actor-critic architecture for automated bilateral negotiations. Specifically, we proposed four different variants of this negotiation model: *ANEGMA*, *ANESIA*, *DLST-ANESIA* and *fuzzy-ANESIA*. In all the variants, we explored the use of DRL with agent technology to help the agent decide what action to take while negotiating with different unknown opponents, which use fixed or adaptive strategies in different negotiation domains. We also explored the use of fuzzy/non-fuzzy multi-objective optimization approaches to generate the Pareto-optimal solution when there is incomplete information about the user's and opponent's preferences. In addition, we used a multiple-criteria decision-making method to select one among many Pareto-optimal solutions generated by a multi-objective optimization approach. Moreover, we used the RECON and GENIUS simulation platforms to perform extensive experimental evaluation.

More specifically, the thesis started by presenting *ANEGMA* in Chapter 3, a novel agent negotiation model supporting agent learning and adaptation during concurrent bilateral negotiations for e-markets like E-bay. An *ANEGMA* agent derives an initial neural network strategy via supervised learning from well-known negotiation models, and evolves the strategy via DRL. We empirically evaluated the performance of an *ANEGMA* buyer agent against fixed but unknown to the agent seller strategies in different e-market settings. We showed that *ANEGMA* outperforms well-known "teacher strategies", the strategies trained with SL only and those trained with DRL only. Crucially, our model also exhibits adaptive behaviour in that it can transfer to environments with unknown sellers, viz., sellers that use different strategies from those used during training.

The thesis continued with introducing *ANESIA* in Chapter 4, another agent model encapsulating different types of learning to support negotiation over multiple issues and under user preference uncertainty. An *ANESIA* agent exploited the notion of "strategy templates" to learn the best combination of acceptance and bidding tactics during negotiation. The model relied on a meta-heuristic approach to learn the tactic choice parameters, and among its tactics, it estimated an adaptive target threshold utility learned with the help of an actor-critic DRL algorithm. Also, *ANESIA* agent used stochastic search based on a nature-inspired single-objective, population-based meta-heuristic approach, called Firefly Algorithm, for user modelling. Moreover, we combined a multi-objective, population-based meta-heuristic approach called NSGA-II and a multiple-criteria decision-making method called TOPSIS for generating Pareto bids during negotiation. We empirically evaluated the performance of *ANESIA* against the winning agent strategies of previous ANAC tournaments in different settings, showing that our agent outperforms opponents known at training time and can effectively transfer its knowledge to environments with previously unseen opponent agents and domains.

We, then presented *DLST-ANESIA*, an extension of *ANESIA* in Chapter 5. *DLST-ANESIA* used the same notion of "strategy templates" as *ANESIA* in the previous chapter. However, *DLST-ANESIA* employed an actor-critic DRL algorithm throughout the strategy, i.e, to learn (a) the tactic choice parameters for acceptance and bidding strategies, and (b) the dynamic threshold utility value. The use of DRL for learning choice parameters in *DLST-ANESIA* allowed the agent accumulate and transfer the knowledge from one negotiation setting to another. This contrasted with *ANESIA*, which used a meta-heuristic approach to estimate choice tactic parameters only once and used the same strategy in all the negotiation settings. Extensive experiments showed that *DLST-ANESIA* agent outperforms *ANESIA* and other agents in terms of individual and social efficiency.

We finally presented *fuzzy-ANESIA*, another extension of *ANESIA* in Chapter 6, which proposed the use of two-phase Pareto-bid generation process during the bidding phase of the negotiation. This chapter, in particular, involves the experimental analysis of a fuzzy-NSGA-II and fuzzy-TOPSIS for the generation of (near) Pareto-optimal bids under user and opponent preference uncertainties. To the best of our knowledge, this combination was the first attempt for solving the multi-objective problem of finding the Pareto-optimal outcomes in multi-issue bilateral negotiations. Extensive experiments showed that the proposed hybrid approach outperforms the other agents in the analysis, as well as the original non-fuzzy Pareto approach.

7.2 Main Findings

The aim of the thesis, which was stated in the Chapter 1, was *to design a learnable negotiation model using deep reinforcement learning for concurrent and non-concurrent bilateral negotiations over one or more issues.* More specifically, the thesis work was designed to address the following research question: "*How can agents learn a negotiation strategy from experience in settings varying from negotiating against different opponent agents to negotiating in different domains?*" This research question was further broken down into various concrete research questions. Below, we summarize how we addressed all these questions:

- *Which DRL algorithm an agent should employ to learn the negotiation strategy? Should the chosen algorithm work for both discrete and continuous action spaces? Can the proposed work be used for both single and multiple issues?* Since there are a number of learning algorithms available in the literature, deciding an appropriate RL algorithm was one of the most difficult tasks. Taking into account the number and type of issues (such as discrete and continuous) and the properties of environment (such as fully-observable, and dynamic), we made the decision of choosing the DDPG algorithm which fitted well according to our assumptions that we made about the negotiation environment. Also, we were interested in the DDPG algorithm because it generates a deterministic action selection policy for the negotiating agent [86]. In addition, it is a *model-free* RL approach, which lets our agent decide what action to take next in a negotiation dialogue, rather than predicting the new state of the environment which was beyond the scope of this work. Moreover, DDPG is an *off-policy* approach, which lets our agent learn a policy/strategy and is different from the one used to take an action. This allowed our agent to do independent exploration of continuous action spaces [86].

- *How can the current state of the negotiation environment be represented?* The choice of input state attributes in every RL algorithm is very important to get an appropriate output value. In our work, we chose a set of different state attributes for each DDPG algorithm, involving the following common attributes: information about the negotiation domain (such as number of issues), preferences of the user (such as initial price and reservation price), and the history of opponent bids (such as last received bid utility and average of all bids received so far).

- *Should the agent negotiate against opponent agents with fixed or dynamic strategies during the learning process? Can we develop a generalized negotiation strategy which is domain-independent as well as opponent-independent? Can the resulting negotiation strategy be interpretable?* In our work, we were interested in an agent strategy which is adaptive, and hence should be learned to negotiate against unseen opponents and in unseen domains (i.e., not seen during the training) efficiently. We were equally interested in a strategy that has the potential to be interpretable, so we proposed the notion of "strategy templates" (see Chapter 4) that consist of a set of negotiation tactics to decide which action to take. Our agent learned the strategy parameters in two different ways: (a) using a meta-heuristic approach by learning the tactic choice parameter values offline and using it in all the negotiation settings, and (b) using an online DRL process, which could adapt the acceptance and bidding strategy while negotiating against different opponents and in different domains. We observed in Chapter 5 that the latter outperformed the former.

- *How can we learn the preferences of an unknown opponent agent during the negotiation?* We used an existing distribution-based frequency model to learn the characteristics of the opponent agent, and showed the increase in negotiation performance with opponent modelling in terms of social welfare utility rate. Proposing a strategy for the opponent model independently was beyond the scope of this thesis.

- *How can we estimate the preferences of the user if only partial information is given to the agent before the negotiation begins?* We introduced the idea of using a nature-inspired meta-heuristic approach for user modelling approaches. However, which meta-heuristic approach to use is negotiable in this thesis, and hence they are treated as pluggable components and can be used according to the application needs. In Chapters 4 and 5, we used the Firefly algorithm, whereas, in Chapter 6, we evaluated the performance of our model using the Cuckoo Search algorithm.

- *How can we reach Pareto-optimal agreements under incomplete information of negotiating parties? How do we deal with the uncertainty in the estimated user and opponent models during the negotiation process?* To reach a Pareto-optimal outcome, our agent had to consider the preferences of both the user agent and the opponent agent, which makes it a multi-objective optimization problem. Since meta-heuristics can provide acceptable solutions in a reasonable time for solving complex problems by efficiently exploring the search space [138], we proposed the use of multi-objective optimization (MOO) method. Since, the MOE method generates a list of Pareto-optimal bids, we combined it with a multi-criteria decision-making (MCDM) method to select one bid among many Pareto optimal bids during the bidding phase of the negotiation. Since we were dealing with incomplete information of the user and opponent utility models, to address the uncertainty while generating the Pareto-optimal bids, we advocated the use of fuzzy MOO and fuzzy MCDM techniques in Chapter 6, with very encouraging results.

- *What performance measures do we need to use to evaluate the decision-making process?* In order to evaluate the overall performance of negotiation, we considered the widely adopted following metrics [88, 2]: average number of rounds, average negotiation time, average distance to the Pareto Curve, average individual utility gained by an agent, average social-welfare utility, and proportion of successful negotiations.

7.3 Contributions

The main contributions of the research work were presented in Chapter 1. For completeness, these are reiterated here, as follows:

- *agent learn-ability*- by proposing four different variants of DRL-based agent negotiation model for automated bilateral negotiations;

- *concurrent negotiations*- by allowing an agent to negotiate with different multiple agents at the same time over one or more issues;

- *agent adaptiveness*- by introducing the notion of "strategy templates" to learn the best combination of acceptance and bidding tactics at any negotiation phase while negotiating against different opponent agents in different domains;

- *user preference modelling*- by proposing to use stochastic search-based approach for estimating the partial preferences of human users which are submitted to the agent before the negotiation begins;

- *social-welfare utility*- by exploring the use of a combination of non-fuzzy/fuzzy-based Multi-Objective Optimization (MOO) algorithm and Multi-Criteria Decision-Making (MCDM) method to generate (near) Pareto-optimal bids and to address the uncertainties in the estimated user and opponent models.

7.4 Future Directions

The research described in this thesis has demonstrated the capability of DRL to let agents learn negotiation strategies that are adaptive and work well against a number of different opponent agents and in different negotiation domains. However, the reported research has also laid the ground for some future research work, as discussed below:

- *User Preference Elicitation:* Although we considered user preference uncertainties in this thesis, we abstracted away from the user preference elicitation problem because it requires multiple interactions with the user. However, preference elicitation is important for obtaining an accurate user model, but may result in user displeasure and bother, especially if the outcome is too large to elicit in its entirety [9]. The open problem here is to explore the use of DRL approaches (i.e., learning based on experience) to consider the trade-off between a good negotiation outcome and the effort required in the elicitation process.

- *Preferential Dependency:* We have assumed independent issues, where the choice of one issue value doesn't have any impact on the choice of other issue's value.

However, often preferential dependencies [95] are ubiquitous in real-life negotiations. To address these preferential dependencies, utility functions may have a more convoluted, non-linear structure, that we leave for future researchers to consider during multi-issue negotiation.

- *Opponent Modelling:* We have also used the existing opponent modelling approach to estimate the characteristics of opponent's behaviour. There are many learning-based approaches proposed by numerous researchers in the literature [59, 25, 154, 158, 84], but, we have been more concerned with learning from experience. So, the open problem here is to amalgamate the use of actor-critic-based learning and opponent modelling that may improve the negotiation performance.

- *Library of Tactics:* In our work, we proposed the concept of "strategy templates" consisting of a series of handcrafted tactics to decide an action. We chose few tactics for acceptance and bidding strategy as per our educated guess, while being mindful of what information is available to the agent from the (external) world/environment or (internal) knowledge base. The possibility of having different tactics/heuristics for each strategy brings up an idea of accommodating a new component in the *ANESIA* model called "Library of Tactics". This could allow developers to design or introduce tactics as they see fit and plug them in their strategies as per their needs. However, how such tactics can be combined, and how their parameters can be learned, is an open problem.

- *Human-Agent Negotiation:* Finally, we have only considered the negotiation between artificial agents. However, in the future, agents will have to negotiate with humans in the real-world, where not all humans will be able to deploy agents to negotiate on their behalf. This would introduce more challenges as the agent would also need to understand emotional and cultural intelligence of humans during negotiation.

Bibliography

[1] Bedour Alrayes, Özgür Kafalı, and Kostas Stathis. RECON: A robust multi-agent environment for simulating concurrent negotiations. In *Recent advances in agent-based complex automated negotiation*, pages 157–174. Springer, 2016.

[2] Bedour Alrayes, Özgür Kafalı, and Kostas Stathis. Concurrent bilateral negotiation for open e-markets: the conan strategy. *Knowledge and Information Systems*, 56(2):463–501, 2018.

[3] Bedour Alrayes and Kostas Stathis. An agent architecture for concurrent bilateral negotiations. In *Decision Support Systems III-Impact of Decision Support Systems for Global Environments*, pages 79–89. Springer, 2013.

[4] Faisal Alsrheed, Abdennour El Rhalibi, Martin Randles, and Madjid Merabti. Intelligent agents for automated cloud computing negotiation. In *2014 international conference on Multimedia Computing and Systems (ICMCS)*, pages 1169–1174. IEEE, 2014.

[5] Bo An and Victor Lesser. Yushu: A heuristic-based agent for automated negotiating competition. In *New Trends in Agent-Based Complex Automated Negotiations*, pages 145–149. Springer, 2012.

[6] Reyhan Aydoğan, Tim Baarslag, Katsuhide Fujita, Johnathan Mell, Jonathan Gratch, Dave De Jonge, Yasser Mohammad, Shinji Nakadai, Satoshi Morinaga, Hirotaka Osawa, et al. Challenges and main results of the automated negotiating agents competition (ANAC) 2019. In *Multi-agent systems and agreement technologies*, pages 366–381. Springer, 2020.

[7] Reyhan Aydoğan, Katsuhide Fujita, Tim Baarslag, Catholijn M Jonker, and Takayuki Ito. ANAC 2017: Repeated multilateral negotiation league. In *International Workshop on Agent-Based Complex Automated Negotiation*, pages 101–115. Springer, 2018.

[8] Tim Baarslag, Katsuhide Fujita, Enrico H Gerding, Koen Hindriks, Takayuki Ito, Nicholas R Jennings, Catholijn Jonker, Sarit Kraus, Raz Lin, Valentin Robu, et al. Evaluating practical negotiating agents: Results and analysis of the 2011 international competition. *Artificial Intelligence*, 198:73–103, 2013.

[9] Tim Baarslag and Enrico H Gerding. Optimal incremental preference elicitation during negotiation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[10] Tim Baarslag, Mark JC Hendrikx, Koen V Hindriks, and Catholijn M Jonker. Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, 30(5):849–898, 2016.

[11] Tim Baarslag, Koen Hindriks, Mark Hendrikx, Alexander Dirkzwager, and Catholijn Jonker. Decoupling negotiating agents to explore the space of negotiation strategies. In *Novel Insights in Agent-based Complex Automated Negotiation*, pages 61–83. Springer, 2014.

[12] Tim Baarslag, Koen Hindriks, Catholijn Jonker, Sarit Kraus, and Raz Lin. The first automated negotiating agents competition (ANAC 2010). In *New Trends in agent-based complex automated negotiations*, pages 113–135. Springer, 2012.

[13] Tim Baarslag and Michael Kaisers. The value of information in automated negotiation: A decision model for eliciting user preferences. In *Proceedings of the 16th conference on autonomous agents and multiagent systems*, pages 391–400, 2017.

[14] Tim Baarslag, Michael Kaisers, Enrico H. Gerding, Catholijn M. Jonker, and Jonathan Gratch. When will negotiation agents be able to represent us? the challenges and opportunities for autonomous negotiators. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4684–4690, 2017.

[15] Pallavi Bagga, Nicola Paoletti, Bedour Alrayes, and Kostas Stathis. Anegma: an automated negotiation model for e-markets. *Autonomous Agents and Multi-Agent Systems*, 35(2):1–28, 2021.

[16] Pallavi Bagga, Nicola Paoletti, and Kostas Alrayes, Bedour Stathis. A deep reinforcement learning approach to concurrent bilateral negotiation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 297–303, 2020.

[17] Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. Learnable strategies for bilateral agent negotiation over multiple issues. *arXiv preprint arXiv:2009.08302*, 2020.

[18] Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. Pareto bid estimation for multi-issue bilateral negotiation under user preference uncertainty. In *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE, 2021.

[19] Oumayma Bahri, El-Ghazali Talbi, and Nahla Ben Amor. A generic fuzzy approach for multi-objective optimization under uncertainty. *Swarm and Evolutionary Computation*, 40:166–183, 2018.

[20] Jasper Bakker, Aron Hammond, Daan Bloembergen, and Tim Baarslag. Rl-boa: A modular reinforcement learning framework for autonomous negotiating agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 260–268. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[21] Mohammad Irfan Bala, Sheetal Vij, and Debajyoti Mukhopadhyay. Intelligent agent for prediction in e-negotiation: an approach. In *2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies*, pages 183–187. IEEE, 2013.

[22] Ken Binmore and Nir Vulkan. Applying game theory to automated negotiation. *Netnomics*, 1(1):1–9, 1999.

[23] Peter Braun, Jakub Brzostowski, Gregory Kersten, Jin Baek Kim, Ryszard Kowalczyk, Stefan Strecker, and Rustam Vahidov. E-negotiation systems and software agents: Methods, models, and applications. In *Intelligent decision-making support systems*, pages 271–300. Springer, 2006.

[24] Joost Broekens, Catholijn M Jonker, and John-Jules Ch Meyer. Affective negotiation support systems. *Journal of Ambient Intelligence and Smart Environments*, 2(2):121–144, 2010.

[25] Scott Buffett and Bruce Spencer. A bayesian classifier for learning opponents' preferences in multi-object automated negotiation. *Electronic Commerce Research and Applications*, 6(3):274–284, 2007.

[26] HH Bui, S Venkatesh, and D Kieronska. An architecture for negotiating agents that learn. Technical report, Citeseer, 1995.

[27] Hung Hai Bui, D Kieronska, and Svetha Venkatesh. Learning other agents' preferences in multiagent negotiation. In *AAAI/IAAI, Vol. 1*, pages 114–119, 1996.

[28] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. Multi-agent reinforcement learning: An overview. *Innovations in multi-agent systems and applications-1*, pages 183–221, 2010.

[29] Réal Carbonneau, Gregory E Kersten, and Rustam Vahidov. Predicting opponent's moves in electronic negotiations using neural networks. *Expert Systems with Applications*, 34(2):1266–1273, 2008.

[30] Henrique Lopes Cardoso and Eugenio Oliveira. Using and evaluating adaptive agents for electronic commerce negotiation. In *Advances in Artificial Intelligence*, pages 96–105. Springer, 2000.

[31] Ho-Chun Herbert Chang. Multi-issue negotiation with deep reinforcement learning. *Knowledge-Based Systems*, page 106544, 2020.

[32] Chen-Tung Chen. Extensions of the topsis for group decision-making under fuzzy environment. *Fuzzy sets and systems*, 114(1):1–9, 2000.

[33] Lihong Chen, Hongbin Dong, Qilong Han, and Guangzhe Cui. Bilateral multi-issue parallel negotiation model based on reinforcement learning. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 40–48. Springer, 2013.

[34] Lihong Chen, Hongbin Dong, and Yang Zhou. A reinforcement learning optimized negotiation method based on mediator agent. *Expert systems with applications*, 41(16):7630–7640, 2014.

[35] Chi-Bin Cheng, Chu-Chai Henry Chan, and Kun-Cheng Lin. Intelligent agents for e-marketplace: Negotiation with issue trade-offs by fuzzy inference systems. *Decision Support Systems*, 42(2):626–638, 2006.

[36] Shi Cheng, Yuhui Shi, and Quande Qin. On the performance metrics of multiobjective optimization. In *International Conference in Swarm Intelligence*, pages 504–512. Springer, 2012.

[37] Samuel PM Choi, Jiming Liu, and Sheung-Ping Chan. A genetic agent-based negotiation system. *Computer Networks*, 37(2):195–204, 2001.

[38] Dave De Jonge and Carles Sierra. Gangster: an automated negotiator applying genetic algorithms. In *Recent advances in agent-based complex automated negotiation*, pages 225–234. Springer, 2016.

[39] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[40] Harri Ehtamo, Raimo P Hämäläinen, Pirja Heiskanen, Jeffrey Teich, Markku Verkama, and Stanley Zionts. Generating pareto solutions in a two-party setting: constraint proposal methods. *Management Science*, 45(12):1697–1709, 1999.

[41] Walaa H El-Ashmawi, Diaa Salama Abd Elminaam, Ayman M Nabil, and Esraa Eldesouky. A chaotic owl search algorithm based bilateral negotiation model. *Ain Shams Engineering Journal*, 11(4):1163–1178, 2020.

[42] Peyman Faratin, Carles Sierra, and Nick R Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3-4):159–182, 1998.

[43] S Shaheen Fatima, Michael Wooldridge, and Nicholas R Jennings. Optimal negotiation strategies for agents with incomplete information. In *International Workshop on Agent Theories, Architectures, and Languages*, pages 377–392. Springer, 2001.

[44] S Shaheen Fatima, Michael J Wooldridge, and Nicholas R Jennings. Multi-issue negotiation with deadlines. *Journal of Artificial Intelligence Research*, 27:381–417, 2006.

[45] Shaheen Fatima, Michael Wooldridge, and Nicholas R Jennings. Optimal negotiation of multiple issues in incomplete information settings. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1080–1087. IEEE Computer Society, 2004.

[46] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. An agenda-based framework for multi-issue negotiation. *Artificial Intelligence*, 152(1):1–45, 2004.

[47] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. A comparative study of game theoretic and evolutionary models of bargaining for software agents. *Artificial Intelligence Review*, 23(2):187–205, 2005.

[48] Tim Frank. Bargaining for advantage: Negotiation strategies for reasonable people. *The Journal of Personal Selling & Sales Management*, 21(4):323, 2001.

[49] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[50] Tomoya Fukui, Ahmed Moustafa, and Takayuki Ito. Jupiter: An automated negotiation environment for supporting agents that use machine learning. In *2018 Thirteenth International Conference on Knowledge, Information and Creativity Support Systems (KICSS)*, pages 1–6. IEEE, 2018.

[51] Michel Gendreau, Jean-Yves Potvin, et al. *Handbook of metaheuristics*, volume 2. Springer, 2010.

[52] Fred Glover. Tabu search and adaptive memory programming—advances, applications and challenges. In *Interfaces in computer science and operations research*, pages 1–75. Springer, 1997.

[53] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016.

[54] Jianye Hao and Ho-Fung Leung. Abines: An adaptive bilateral negotiating strategy over multiple items. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 95–102. IEEE, 2012.

[55] Jianye Hao, Songzheng Song, Ho-fung Leung, and Zhong Ming. An efficient and robust negotiating strategy in bilateral negotiations over multiple items. *Engineering Applications of Artificial Intelligence*, 34:45–57, 2014.

[56] Keisuke Hara and Takayuki Ito. A mediation mechanism for automated negotiating agents whose utility changes over time. In *Twenty-seventh AAAI conference on artificial intelligence*, 2013.

[57] Khayyam Hashmi, Amal Alhosban, Erfan Najmi, Zaki Malik, et al. Automated web service quality component negotiation using NSGA-2. In *2013 ACS International Conference on Computer Systems and Applications (AICCSA)*, pages 1–6. IEEE, 2013.

[58] Pirja Heiskanen, Harri Ehtamo, and Raimo P Hämäläinen. Constraint proposal method for computing pareto solutions in multi-party negotiations. *European Journal of Operational Research*, 133(1):44–61, 2001.

[59] Koen Hindriks and Dmytro Tykhonov. Opponent modelling in automated multi-issue negotiation using bayesian learning. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 331–338, 2008.

[60] Koen V Hindriks and Catholijn M Jonker. Creating human-machine synergy in negotiation support systems: Towards the pocket negotiator. In *Proceedings of the 1st International Working Conference on Human Factors and Computational Models in Negotiation*, pages 47–54, 2008.

[61] Holger H Hoos and Thomas Stützle. *Stochastic local search: Foundations and applications*. Elsevier, 2004.

[62] Chongming Hou. Predicting agents tactics in automated negotiation. In *Proceedings. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004.(IAT 2004).*, pages 127–133. IEEE, 2004.

[63] Ching-Lai Hwang, Young-Jou Lai, and Ting-Yun Liu. A new approach for multiple objective decision making. *Computers & operations research*, 20(8):889–899, 1993.

[64] Ching-Lai Hwang and Kwangsun Yoon. Methods for multiple attribute decision making. In *Multiple attribute decision making*, pages 58–191. Springer, 1981.

[65] Kashif Imran, Jiangfeng Zhang, Anamitra Pal, Abraiz Khattak, Kafait Ullah, and Sherjeel Mahmood Baig. Bilateral negotiations for electricity market by adaptive agent-tracking strategy. *Electric Power Systems Research*, 186:106390, 2020.

[66] Hamid Jazayeriy, Masrah Azmi-Murad, Nasir Sulaiman, and Nur Izura Udizir. Pareto-optimal algorithm in bilateral automated negotiation. *International Journal of Digital Content Technology and its Applications*, 5(3):1–11, 2011.

[67] Hamid Jazayeriy, Masrah Azmi-Murad, Nasir Sulaiman, and Nur Izura Udzir. Generating pareto-optimal offers in bilateral automated negotiation with one-side uncertain importance weights. *Computing and Informatics*, 31(6):1235–1253, 2013.

[68] Li Jian. An agent bilateral multi-issue alternate bidding negotiation protocol based on reinforcement learning and its application in e-commerce. In *2008 International Symposium on Electronic Commerce and Security*, pages 217–220. IEEE, 2008.

[69] Dave de Jonge and Carles Sierra. D-brane: a diplomacy playing agent for automated negotiations research. *Applied Intelligence*, 47(1):158–177, 2017.

[70] Catholijn Jonker, Reyhan Aydogan, Tim Baarslag, Katsuhide Fujita, Takayuki Ito, and Koen Hindriks. Automated negotiating agents competition (anac). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[71] Catholijn M Jonker, Reyhan Aydoğan, Tim Baarslag, Joost Broekens, Christian A Detweiler, Koen V Hindriks, Alina Huldtgren, and Wouter Pasman. An introduction to the pocket negotiator: a general purpose negotiation support

system. In *Multi-Agent Systems and Agreement Technologies*, pages 13–27. Springer, 2016.

[72] Catholijn M Jonker and Takayuki Ito. Anac 2018: Repeated multilateral negotiation league. In *Advances in Artificial Intelligence: Selected Papers from the Annual Conference of Japanese Society of Artificial Intelligence (JSAI 2019)*, volume 1128, page 77. Springer Nature, 2020.

[73] Catholijn M Jonker, Valentin Robu, and Jan Treur. An agent architecture for multi-attribute negotiation using incomplete preference information. *Autonomous Agents and Multi-Agent Systems*, 15(2):221–252, 2007.

[74] Yoshiaki Kadono. Agent yk: An efficient estimation of opponent's intention with stepped limited concessions. In *Recent advances in agent-based complex automated negotiation*, pages 279–283. Springer, 2016.

[75] James Kennedy. Swarm intelligence. In *Handbook of nature-inspired and innovative computing*, pages 187–219. Springer, 2006.

[76] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[77] Usha Kiruthika, Thamarai Selvi Somasundaram, and S Kanaga Suba Raja. Lifecycle model of a negotiation agent: A survey of automated negotiation techniques. *Group Decision and Negotiation*, pages 1–24, 2020.

[78] Mark Klein, Peyman Faratin, Hiroki Sayama, and Yaneer Bar-Yam. Negotiating complex contracts. *Group Decision and Negotiation*, 12(2):111–125, 2003.

[79] Elias Kougianos and Saraju P Mohanty. A nature-inspired firefly algorithm based approach for nanoscale leakage optimal rtl structure. *Integration, the VLSI Journal*, 51:46–60, 2015.

[80] Dan E Kröhling, Omar JA Chiotti, and Ernesto C Martínez. A context-aware approach to automated negotiation using reinforcement learning. *Advanced Engineering Informatics*, 47:101229, 2021.

[81] Guoming Lai, Cuihong Li, Katia Sycara, and Joseph Giampapa. Literature review on multi-attribute negotiations. *Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-RI-TR-04-66*, 2004.

[82] Raymond YK Lau, Maolin Tang, On Wong, Stephen W Milliner, and Yi-Ping Phoebe Chen. An evolutionary learning approach for adaptive negotiation agents. *International Journal of Intelligent Systems*, 21(1):41–72, 2006.

[83] Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125*, 2017.

[84] Jian Li and Yuan-Da Cao. Bayesian learning in bilateral multi-issue negotiation and its application in mas-based electronic commerce. In *Proceedings. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004.(IAT 2004).*, pages 437–440. IEEE, 2004.

[85] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[86] Timothy Paul Lillicrap, Jonathan James Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, Yuval Tassa, David Silver, and Daniel Pieter Wierstra. Continuous control with deep reinforcement learning, January 26 2017. US Patent App. 15/217,758.

[87] Raz Lin and Sarit Kraus. Can automated agents proficiently negotiate with humans? *Communications of the ACM*, 53(1):78–88, 2010.

[88] Raz Lin, Sarit Kraus, Tim Baarslag, Dmytro Tykhonov, Koen Hindriks, and Catholijn M Jonker. Genius: An integrated environment for supporting the

design of generic automated negotiators. *Computational Intelligence*, 30(1):48–70, 2014.

[89] Raz Lin, Sarit Kraus, Jonathan Wilkenfeld, and James Barry. Negotiating with bounded rational agents in environments with incomplete information using an automated agent. *Artificial Intelligence*, 172(6-7):823–851, 2008.

[90] Alessio R Lomuscio, Michael Wooldridge, and Nicholas R Jennings. A classification scheme for negotiation in electronic commerce. *Group Decision and Negotiation*, 12(1):31–56, 2003.

[91] Fernando Lopes, Michael Wooldridge, and Augusto Q Novais. Negotiation among autonomous computational agents: principles, analysis and challenges. *Artificial Intelligence Review*, 29(1):1–44, 2008.

[92] Xudong Luo, Nicholas R Jennings, Nigel Shadbolt, Ho-fung Leung, and Jimmy Ho-man Lee. A fuzzy constraint based model for bilateral, multi-issue negotiations in semi-competitive environments. *Artificial Intelligence*, 148(1-2):53–102, 2003.

[93] Abbas Mardani, Ahmad Jusoh, Khalil Nor, Zainab Khalifah, Norhayati Zakwan, and Alireza Valipour. Multiple criteria decision-making techniques and their applications–a review of the literature from 2000 to 2014. *Economic research-Ekonomska istraživanja*, 28(1):516–571, 2015.

[94] Ivan Marsa-Maestre, Mark Klein, Enrique de la Hoz, and Miguel A Lopez-Carmona. Negowiki: A set of community tools for the consistent comparison of negotiation approaches. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 424–435. Springer, 2011.

[95] Ivan Marsa-Maestre, Mark Klein, Catholijn M Jonker, and Reyhan Aydoğan. From problems to protocols: Towards a negotiation handbook. *Decision Support Systems*, 60:39–54, 2014.

[96] Ivan Marsa-Maestre, Miguel A Lopez-Carmona, Juan R Velasco, Takayuki Ito, Mark Klein, and Katsuhide Fujita. Balancing utility and deal probability for auction-based negotiations in highly nonlinear utility spaces. In *Twenty-first international joint conference on artificial intelligence*, 2009.

[97] Johnathan Mell and Jonathan Gratch. Iago: interactive arbitration guide online. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1510–1512, 2016.

[98] Melanie Mitchell. *An introduction to genetic algorithms.* MIT press, 1998.

[99] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[100] Mina Montazeri, Hamed Kebriaei, and Babak N Araabi. Learning pareto optimal solution of a multi-attribute bilateral negotiation using deep reinforcement. *Electronic Commerce Research and Applications*, 43:100987, 2020.

[101] Ataul Munim. *GOLEMLite: A framework for the development of agent-based applications.* Master's Thesis, Royal Holloway, University of London, 2013.

[102] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[103] Vidya Narayanan and Nicholas R Jennings. Learning to negotiate optimally in non-stationary environments. In *International Workshop on Cooperative Information Agents*, pages 288–300. Springer, 2006.

[104] A Nazeri and M Bafrouei. A fuzzy NSGA-II for supplier selection and multi-product allocation order. *Uncertain Supply Chain Management*, 3(3):241–252, 2015.

[105] Thuc Duong Nguyen and Nicholas R Jennings. A heuristic model of concurrent bi-lateral negotiations in incomplete information settings. 2003.

[106] Thuc Duong Nguyen and Nicholas R Jennings. Coordinating multiple concurrent negotiations. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1064–1071. IEEE Computer Society, 2004.

[107] M North, P Thimmapuram, R Cirillo, C Macal, G Conzelmann, G Boyd, V Koritarov, and T Veselka. Emcas: An agent-based tool for modeling electricity markets. In *2003 Agent Conference on Challenges in Social Simulation*, page 253, 2003.

[108] Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11(3):387–434, 2005.

[109] Alexandros Papangelis and Kallirroi Georgila. Reinforcement learning of multi-issue negotiation dialogue policies. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 154–158, 2015.

[110] Tiago Pinto, Gabriel Santos, Zita Vale, Isabel Praça, Fernando Lopes, and Hugo Algarvio. Realistic multi-agent simulation of competitive electricity markets. In *2014 25th International Workshop on Database and Expert Systems Applications*, pages 109–113. IEEE, 2014.

[111] Isabel Praça, Carlos Ramos, Zita Vale, and Manuel Cordeiro. Mascem: a multiagent system that simulates competitive electricity markets. *IEEE Intelligent Systems*, 18(6):54–60, 2003.

[112] Iyad Rahwan, Ryszard Kowalczyk, and Ha Hai Pham. Intelligent agents for automated one-to-many e-commerce negotiation. In *Australian Computer Science Communications*, volume 24, pages 197–204. Australian Computer Society, Inc., 2002.

[113] Howard Raiffa. *The art and science of negotiation.* Harvard University Press, 1982.

[114] Nadia Ranaldo and Eugenio Zimeo. Capacity-aware utility function for sla negotiation of cloud services. In *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*, pages 292–296. IEEE, 2013.

[115] Ritu Rawat. *User modelling for multi-issue negotiation: The role of population-based meta-heuristic algorithms over discrete issues.* Master's Thesis, Royal Holloway, University of London, 2021.

[116] Yinon Oshrat Raz Lin and Sarit Kraus. Investigating the benefits of automated negotiations in enhancing people's negotiation skills. In *AAMAS*, volume 9, pages 345–352. Citeseer, 2009.

[117] Yousef Razeghi, Celal Ozan Berk Yavuz, and Reyhan Aydoğan. Deep reinforcement learning for acceptance strategy in bilateral negotiations. *Turkish Journal of Electrical Engineering & Computer Sciences*, 28(4):1824–1840, 2020.

[118] Fenghui Ren, Minjie Zhang, and Quan Bai. A dynamic, optimal approach for multi-issue negotiation under time constraints. In *Novel insights in agent-based complex automated negotiation*, pages 85–108. Springer, 2014.

[119] Zhaomin Ren and Chimay J Anumba. Learning in multi-agent systems: a case study of construction claims negotiation. *Advanced engineering informatics*, 16(4):265–275, 2002.

[120] Ariel Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109, 1982.

[121] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern approach.* Pearson, 3 edition, 2009.

[122] Sabyasachi Saha, Anish Biswas, and Sandip Sen. Modeling opponent decision in repeated one-shot negotiations. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 397–403, 2005.

[123] Sabyasachi Saha and Sandip Sen. Negotiating efficient outcomes over multiple issues. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 423–425, 2006.

[124] Victor Sanchez-Anguix, Reyhan Aydoğan, Tim Baarslag, and Catholijn Jonker. Bottom-up approaches to achieve pareto optimal agreements in group decision making. *Knowledge and Information Systems*, 61(2):1019–1046, 2019.

[125] Victor Sanchez-Anguix, Reyhan Aydoğan, Tim Baarslag, and Catholijn M Jonker. Can we reach pareto optimal outcomes using bottom-up approaches? In *International Workshop on Conflict Resolution in Decision Making*, pages 19–35. Springer, 2016.

[126] Motoki Sato and Takayuki Ito. Whaleagent: Hardheaded strategy and conceder strategy based on the heuristics. In *Recent advances in agent-based complex automated negotiation*, pages 273–278. Springer, 2016.

[127] Francisco Silva, Ricardo Faia, Tiago Pinto, Isabel Praça, and Zita Vale. Optimizing opponents selection in bilateral contracts negotiation with particle swarm. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 116–124. Springer, 2018.

[128] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[129] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[130] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[131] Kwang Mong Sim, Yuanyuan Guo, and Benyun Shi. Adaptive bargaining agents that negotiate optimally and rapidly. In *2007 IEEE Congress on Evolutionary Computation*, pages 1007–1014. IEEE, 2007.

[132] Kwang Mong Sim, Yuanyuan Guo, and Benyun Shi. BLGAN: Bayesian learning and genetic algorithm for supporting negotiation with incomplete information. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):198–211, 2008.

[133] Manu Sridharan and Gerald Tesauro. Multi-agent q-learning and regression trees for automated pricing decisions. In *Game Theory and Decision Theory in Agent-Based Systems*, pages 217–234. Springer, 2002.

[134] Tianhao Sun, Qingsheng Zhu, Yunni Xia, and Feng Cao. A bilateral price negotiation strategy based on bayesian classification and q-learning. *Journal of Information and Computational Science*, 8(13):2773–2780, 2011.

[135] Richard S Sutton and Andrew G Barto. *Introduction to Reinforcement Learning*. MIT press Cambridge, 1998.

[136] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[137] Katia Sycara, Daniel Zeng, et al. Benefits of learning in negotiation. In *Proceedings of the AAAI National Conference on Artificial Intelligence. Menlo Park, California*, pages 36–41, 1997.

[138] El-Ghazali Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009.

[139] Gerald Tesauro and Jeffrey O Kephart. Pricing in agent economies using multi-agent q-learning. *Autonomous agents and multi-agent systems*, 5(3):289–304, 2002.

[140] Leigh L Thompson and Leigh L Thompson. *The mind and heart of the negotiator*. Pearson/Prentice Hall Upper Saddle River, NJ, 2005.

[141] Dimitrios Tsimpoukis, Tim Baarslag, Michael Kaisers, and Nikolaos G Paterakis. Automated negotiations under user preference uncertainty: A linear programming approach. In *International conference on agreement technologies*, pages 115–129. Springer, 2018.

[142] Okan Tunalı, Reyhan Aydoğan, and Victor Sanchez-Anguix. Rethinking frequency opponent modeling in automated negotiation. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 263–279. Springer, 2017.

[143] Gwo-Hshiung Tzeng and Jih-Jeng Huang. *Multiple attribute decision making: methods and applications*. CRC press, 2011.

[144] Tomasz Wachowicz, Gregory E Kersten, and Ewa Roszkowska. How do I tell you what I want? Agent's interpretation of principal's preferences and its impact on understanding the negotiation process and outcomes. *Operational Research*, 19(4):993–1032, 2019.

[145] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[146] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge United Kingdom, 1989.

[147] Gerhard Weiß. Adaptation and learning in multi-agent systems: Some remarks and a bibliography. In *International Joint Conference on Artificial Intelligence*, pages 1–21. Springer, 1995.

[148] Colin R Williams, Valentin Robu, Enrico H Gerding, and Nicholas R Jennings. Negotiating concurrently with unknown opponents in complex, real-time domains. In *Proceedings of the 20th European Conference on Artificial Intelligence*, 2012.

[149] Colin R Williams, Valentin Robu, Enrico H Gerding, and Nicholas R Jennings. An overview of the results and insights from the third automated negotiating agents competition (anac2012). *Novel Insights in Agent-based Complex Automated Negotiation*, pages 151–162, 2014.

[150] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[151] Xin-She Yang. Firefly algorithms for multimodal optimization. In *International symposium on stochastic algorithms*, pages 169–178. Springer, 2009.

[152] Xin-She Yang. *Nature-inspired algorithms and applied optimization*, volume 744. Springer, 2017.

[153] Xin-She Yang and Suash Deb. Cuckoo search via lévy flights. In *2009 World congress on nature & biologically inspired computing (NaBIC)*, pages 210–214. Ieee, 2009.

[154] Chao Yu, Fenghui Ren, and Minjie Zhang. An adaptive bilateral negotiation model based on bayesian learning. In *Complex automated negotiations: Theories, models, and software competitions*, pages 75–93. Springer, 2013.

[155] Farhad Zafari, Mofakham Faria Nassiri, and Hamadani Ali Zeinal. Dopponent: A socially efficient preference model of opponent in bilateral multi issue negotiations. *Journal of Computing and Society*, 2014.

[156] Farhad Zafari and Faria Nassiri-Mofakham. Popponent: Highly accurate, individually and socially efficient opponent preference model in bilateral multi issue negotiations. *Artificial Intelligence*, 237:59–91, 2016.

[157] Dajun Zeng and Katia Sycara. Bayesian learning in negotiation. *International Journal of Human-Computer Studies*, 48(1):125–141, 1998.

[158] Jihang Zhang, Fenghui Ren, and Minjie Zhang. Bayesian-based preference prediction in bilateral multi-issue negotiation between intelligent agents. *Knowledge-Based Systems*, 84:108–120, 2015.

[159] Mingwen Zhang, Zhongfu Tan, Jianbao Zhao, and Li Li. A bayesian learning model in the agent-based bilateral negotiation between the coal producers and electric power generators. In *2008 International Symposium on Intelligent Information Technology Application Workshops*, pages 859–862. IEEE, 2008.

[160] Yi Zou, Wenjie Zhan, and Yuan Shao. Evolution with reinforcement learning in negotiation. *PLOS one*, 9(7):e102840, 2014.

Appendix A

Domains Used for Multi-Issue Negotiation

Various multi-issue domains (all are readily available in GENIUS [88]) which were used during the evaluation of *ANESIA*, *DLST-ANESIA* and *fuzzy-ANESIA* are described as follows:

- **Itex-Cypress:** It is a buyer–seller business negotiation for one commodity. It involves representatives of two companies: Itex Manufacturing, a producer of bicycle components and Cypress Cycles, a builder of bicycles. This domain consists of 4 issues: the price of the components, delivery times, payment arrangements and terms for the return of possibly defective parts. Each issue has 3 or 4 values, resulting in a domain with total 180 possible outcomes.

- **Laptop:** In this domain, a buyer and a seller negotiates over the specification of a laptop. This domain consists of 3 issues: the laptop brand, size of the external monitor and the size of the hard disk. Each issue consists of 3 values, and hence makes it the smallest domain with only 27 possible outcomes.

- **Grocery:** In this domain, two agents representing two people living together negotiates in a local supermarket who have different tastes. The domain consists of 5 types of products (or issues): bread, fruit, snacks, spreads and vegetables. Each category has further 4 or 5 products, resulting in a medium-sized domain with 1600 possible outcomes.

- **Camera:** In this domain, a buyer agent and a seller agent negotiates over a camera. The domain consists of 6 issues: maker, body, lens, tripod, bags and accessories. Each category has 3 to 5 values, resulting in a domain with total 3600 possible outcomes.

- **Energy (or Small Energy):** In this domain, an agent representing electricity distribution company negotiates with another agent representing large consumer. They negotiate over issues representing how much the consumer is willing to reduce its consumption over a number of time slots for a day-ahead market. This domain is the largest with 15625 total possible outcomes.

- **Holiday:** In this domain, two agents representing two people plan for their next holiday while negotiating over the following 5 issues: destination, duration, budget, type of activities involved, and the transportation. Each issue has 4 values, resulting in a domain with 1024 possible outcomes.

- **Party:** In this domain, two agents reprinting two people living together while organizing a party negotiates over 6 issues: the food type, drinks type, location, type of invitations, music and the clean up service. Each issue further consists of 3 to 5 values, resulting in a domain with 3072 total possible outcomes.

- **Smart Energy Grid:** This domain is similar to Energy domain with only 4 issues, each having 5 values. This domain consists of total 625 possible outcomes.

- **Fitness:** In this domain, two agents representing two people negotiate over fitness membership. The domain consists of 5 issues such as: kind of fitness, duration, distance to fitness center venue, fitness intensity, and the membership. Each issue has 4 or 11 values, resulting in a domain with 3520 possible outcomes.

- **Flight Booking:** In this domain, two agents negotiate while booking a flight over 3 issues: ticket price, departure airport city and departure date. Each issue has 3 to 5 values, resulting in the domain with only 48 total possible outcomes.

- **Outfit:** In this domain, two agents negotiate over the dress they should put on for an event. There are 4 issues: shirts, pants, shoes and accessories, each having 2 or 4 values, resulting in a domain with 128 total possible outcomes.

Appendix B

Performance of ANESIA model

Airport Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	119.3 ± 125.27	0.2 ± 0.26	1.38 ± 0.38	0.6 ± 0.2	0.81 ± 0.2	0.95
AgentGP •	127.66 ± 116.52	0.04 ± 0.14	1.48 ± 0.21	0.64 ± 0.12	0.65 ± 0.12	0.99
FSEGA2019 •	439.89 ± 636.24	**0.06 ± 0.14**	1.47 ± 0.18	0.75 ± 0.16	0.75 ± 0.16	**1.00**
AgentHerb ◇	**10.64 ± 7.45**	0.2 ± 0.22	1.25 ± 0.24	0.48 ± 0.22	0.48 ± 0.22	**1.00**
Agent33 ◇	374.62 ± 500.5	0.2 ± 0.24	1.23 ± 0.35	0.54 ± 0.16	0.54 ± 0.16	0.99
Sontag ◇	434.54 ± 488.25	0.11 ± 0.24	1.38 ± 0.34	0.73 ± 0.19	0.74 ± 0.19	0.97
AgreeableAgent ◇	673.59 ± 705.79	0.13 ± 0.27	1.34 ± 0.37	**0.82 ± 0.19**	0.83 ± 0.19	0.96
PonpokoAgent ⋆	688.06 ± 648.93	0.18 ± 0.31	1.29 ± 0.44	**0.82 ± 0.14**	**0.85 ± 0.1**	0.92
ParsCat2 ⋆	492.9 ± 539.28	0.09 ± 0.17	1.42 ± 0.25	0.75 ± 0.19	0.75 ± 0.19	**1.00**

Airport Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	117.9 ± 82.03	0.2 ± 0.22	1.19 ± 0.54	0.58 ± 0.23	0.79 ± 0.14	0.73
AgentGP •	98.23 ± 88.77	0.04 ± 0.10	1.17 ± 0.34	0.72 ± 0.18	0.74 ± 0.17	0.98
FSEGA2019 •	365.7 ± 388.48	0.06 ± 0.09	1.09 ± 0.32	0.74 ± 0.19	0.75 ± 0.18	0.98
AgentHerb ◇	**3.83 ± 1.39**	**0.02 ± 0.05**	**1.52 ± 0.13**	0.60 ± 0.13	0.60 ± 0.13	**1.00**
Agent33 ◇	128.23 ± 215.83	0.06 ± 0.09	1.33 ± 0.19	0.61 ± 0.15	0.61 ± 0.15	1.00
Sontag ◇	363.22 ± 387.03	0.06 ± 0.12	1.09 ± 0.36	0.74 ± 0.16	0.77 ± 0.13	0.96
AgreeableAgent ◇	516.15 ± 483.34	0.05 ± 0.1	0.94 ± 0.36	**0.82 ± 0.18**	0.83 ± 0.15	0.97
PonpokoAgent ⋆	516.42 ± 483.8	0.08 ± 0.14	0.92 ± 0.38	0.80 ± 0.18	**0.84 ± 0.09**	0.93
ParsCat2 ⋆	417.44 ± 424.29	0.06 ± 0.1	1.05 ± 0.33	0.76 ± 0.16	0.77 ± 0.14	0.98

Table B.1: Performance of *ANESIA-Random* - Ablation Study2 - over domain AIRPORT (1440 ×2 profiles = 2880 simulations)

Camera Domain B = 5% of Ω

Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	274.92 ± 264.37	0.29 ± 0.09	**1.19** ± **0.56**	0.65 ± 0.15	0.71 ± 0.14	0.74
AgentGP ●	110.31 ± 240.73	0.16 ± 0.2	1.14 ± 0.42	0.63 ± 0.19	0.65 ± 0.2	0.91
FSEGA2019 ●	58.45 ± 37.13	0.16 ± 0.25	1.1 ± 0.49	0.78 ± 0.15	0.82 ± 0.1	0.86
AgentHerb ◇	**5.15 ± 2.07**	**0.06** ± **0.09**	1.4 ± 0.21	0.52 ± 0.22	0.52 ± 0.22	**1.00**
Agent33 ◇	410.81 ± 630.4	**0.06** ± **0.07**	1.4 ± 0.2	0.7 ± 0.18	0.7 ± 0.18	**1.00**
Sontag ◇	651.37 ± 1116.79	**0.06** ± **0.1**	1.36 ± 0.23	0.74 ± 0.14	0.74 ± 0.14	0.99
AgreeableAgent ◇	1094.92 ± 1708.08	0.09 ± 0.19	1.14 ± 0.38	**0.82** ± **0.17**	0.84 ± 0.15	0.93
PonpokoAgent ⋆	966.69 ± 1428.05	0.17 ± 0.28	1.06 ± 0.54	0.79 ± 0.16	**0.86** ± **0.09**	0.82
ParsCat2 ⋆	727.34 ± 1084.22	0.08 ± 0.15	1.22 ± 0.34	0.72 ± 0.22	0.73 ± 0.22	0.96

Camera Domain B = 10% of Ω

Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	250.76 ± 274.49	0.2 ± 0.18	1.38 ± 0.49	0.56 ± 0.32	0.72 ± 0.12	0.78
AgentGP ●	74.14 ± 210.46	0.14 ± 0.15	1.05 ± 0.48	0.62 ± 0.25	0.71 ± 0.1	0.88
FSEGA2019 ●	50.78 ± 38.82	0.09 ± 0.15	1.03 ± 0.46	0.74 ± 0.28	0.83 ± 0.11	0.89
AgentHerb ◇	**3.14 ± 0.94**	0.06 ± 0.1	**1.40** ± **0.2**	0.51 ± 0.19	0.51 ± 0.19	**1.00**
Agent33 ◇	173.88 ± 189.77	0.05 ± 0.07	1.37 ± 0.2	0.66 ± 0.18	0.66 ± 0.18	**1.00**
Sontag ◇	594.8 ± 919.74	**0.03** ± **0.07**	1.26 ± 0.27	**0.79** ± **0.15**	0.8 ± 0.13	0.99
AgreeableAgent ◇	1013.27 ± 1200.7	0.06 ± 0.13	0.92 ± 0.36	0.78 ± 0.26	0.84 ± 0.17	0.93
PonpokoAgent ⋆	754.45 ± 992.61	0.08 ± 0.17	1.02 ± 0.48	0.76 ± 0.31	**0.88** ± **0.06**	0.87
ParsCat2 ⋆	532.5 ± 745.04	0.05 ± 0.09	1.16 ± 0.34	0.78 ± 0.17	0.80 ± 0.12	0.97

Table B.2: Performance of *ANESIA-Random* - Ablation Study2 - over domain Camera (1440 ×2 profiles = 2880 simulations)

Energy Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	824.42 ± 789.03	0.19 ± 0.21	**1.11 ± 0.29**	0.45 ± 0.22	0.46 ± 0.22	0.94
AgentGP ●	56.54 ± 208.7	0.47 ± 0.38	0.64 ± 0.55	0.42 ± 0.19	0.54 ± 0.15	0.58
FSEGA2019 ●	215.66 ± 125.98	0.6 ± 0.4	0.41 ± 0.53	0.48 ± 0.29	**0.85 ± 0.08**	0.38
AgentHerb ◇	**35.12 ± 25.49**	**0.08 ± 0.08**	1.08 ± 0.1	0.25 ± 0.16	0.25 ± 0.16	1.00
Agent33 ◇	7123.76 ± 9819.55	0.15 ± 0.12	1.10 ± 0.18	0.48 ± 0.17	0.48 ± 0.17	**0.98**
Sontag ◇	9379.17 ± 12799.92	0.37 ± 0.36	0.77 ± 0.51	**0.57 ± 0.23**	0.71 ± 0.12	0.71
AgreeableAgent ◇	10773.89 ± 13829.31	0.23 ± 0.29	0.92 ± 0.39	0.46 ± 0.26	0.5 ± 0.27	0.86
PonpokoAgent ⋆	10507.22 ± 13470.76	0.52 ± 0.42	0.51 ± 0.54	0.53 ± 0.3	0.84 ± 0.1	0.47
ParsCat2 ⋆	7395.98 ± 9537.85	0.37 ± 0.37	0.75 ± 0.51	0.55 ± 0.24	0.69 ± 0.17	0.69

Energy Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	796.79 ± 817.44	0.08 ± 0.09	**1.14 ± 0.27**	0.5 ± 0.21	0.51 ± 0.21	0.95
AgentGP ●	54.79 ± 186.11	0.46 ± 0.39	0.66 ± 0.57	0.43 ± 0.19	0.56 ± 0.14	0.58
FSEGA2019 ●	214.71 ± 125.46	0.59 ± 0.4	0.44 ± 0.53	0.49 ± 0.3	**0.84 ± 0.08**	0.41
AgentHerb ◇	**35.2 ± 26.87**	**0.08 ± 0.08**	1.10 ± 0.11	0.26 ± 0.16	0.26 ± 0.16	**1.00**
Agent33 ◇	6776.28 ± 8751.38	0.16 ± 0.17	1.08 ± 0.25	0.46 ± 0.17	0.47 ± 0.16	0.96
Sontag ◇	8960.3 ± 12934.01	0.36 ± 0.36	0.79 ± 0.5	**0.57 ± 0.23**	0.69 ± 0.13	0.72
AgreeableAgent ◇	9835.89 ± 13329.52	0.2 ± 0.26	0.97 ± 0.35	0.49 ± 0.26	0.52 ± 0.27	0.89
PonpokoAgent ⋆	9539.13 ± 12556.79	0.54 ± 0.41	0.49 ± 0.55	0.52 ± 0.3	0.84 ± 0.09	0.46
ParsCat2 ⋆	7983.96 ± 10401.7	0.38 ± 0.36	0.75 ± 0.51	0.54 ± 0.23	0.67 ± 0.15	0.69

Table B.3: Performance of *ANESIA-Random* - Ablation Study2 - over domain Energy (1440 ×2 profiles = 2880 simulations)

Grocery Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	237.6 ± 225.72	**0.21** ± **0.24**	**1.27** ± **0.45**	0.64 ± 0.28	0.73 ± 0.14	0.87
AgentGP ●	91.72 ± 175.14	0.21 ± 0.15	**1.27** ± **0.31**	**0.72** ± **0.19**	**0.75** ± **0.13**	0.96
FSEGA2019 ●	206.3 ± 292.85	0.22 ± 0.1	1.22 ± 0.22	0.64 ± 0.16	0.65 ± 0.14	0.98
AgentHerb ◇	**4.96** ± **2.37**	0.26 ± 0.07	1.24 ± 0.14	0.51 ± 0.15	0.51 ± 0.15	**1.00**
Agent33 ◇	90.53 ± 97.13	0.24 ± 0.07	1.25 ± 0.14	0.57 ± 0.16	0.57 ± 0.16	**1.00**
Sontag ◇	363.54 ± 669.3	0.21 ± 0.08	1.25 ± 0.17	0.66 ± 0.14	0.66 ± 0.13	0.99
AgreeableAgent ◇	808.19 ± 812.66	0.23 ± 0.11	1.04 ± 0.19	0.68 ± 0.13	0.69 ± 0.09	0.98
PonpokoAgent ⋆	441.17 ± 605.81	0.22 ± 0.12	1.20 ± 0.24	0.68 ± 0.15	0.70 ± 0.1	0.97
ParsCat2 ⋆	313.02 ± 395.48	0.22 ± 0.1	1.23 ± 0.22	0.66 ± 0.16	0.67 ± 0.13	0.98
Grocery Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	262.29 ± 255.59	**0.22** ± **0.22**	1.28 ± 0.36	0.67 ± 0.16	0.68 ± 0.16	0.95
AgentGP ●	134.48 ± 252.2	0.25 ± 0.22	**1.33** ± **0.34**	0.70 ± 0.12	0.71 ± 0.11	0.95
FSEGA2019 ●	291.65 ± 329.34	0.24 ± 0.09	1.32 ± 0.14	0.67 ± 0.11	0.67 ± 0.11	**1.00**
AgentHerb ◇	**8.42** ± **3.82**	**0.22** ± **0.08**	1.31 ± 0.12	0.54 ± 0.1	0.54 ± 0.1	**1.00**
Agent33 ◇	531.96 ± 769.85	0.27 ± 0.14	1.29 ± 0.22	0.63 ± 0.09	0.63 ± 0.09	0.98
Sontag ◇	392.43 ± 571.41	0.23 ± 0.1	**1.33** ± **0.15**	0.65 ± 0.11	0.65 ± 0.11	**1.00**
AgreeableAgent ◇	1090.66 ± 1101.45	0.24 ± 0.12	1.31 ± 0.18	0.71 ± 0.1	0.72 ± 0.09	0.99
PonpokoAgent ⋆	769.14 ± 919.8	0.24 ± 0.13	1.32 ± 0.19	**0.74** ± **0.1**	**0.74** ± **0.1**	0.98
ParsCat2 ⋆	598.63 ± 621.81	0.25 ± 0.1	1.31 ± 0.16	0.69 ± 0.09	0.69 ± 0.09	0.99

Table B.4: Performance of *ANESIA-Random* - Ablation Study2 - over domain Grocery (1440 ×2 profiles = 2880 simulations)

Fitness Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	7.11 ± 2.59	0.05 ± 0.07	**1.49 ± 0.08**	0.89 ± 0.13	**0.89 ± 0.13**	**1.00**
AgentGP ●	577.86 ± 1122.43	0.19 ± 0.31	1.16 ± 0.56	0.64 ± 0.16	0.67 ± 0.16	0.82
FSEGA2019 ●	1476.17 ± 1984.41	0.1 ± 0.16	1.3 ± 0.29	0.77 ± 0.13	0.78 ± 0.12	0.96
AgentHerb ◇	**7.23 ± 3.22**	**0.04 ± 0.05**	**1.49 ± 0.05**	0.58 ± 0.1	0.58 ± 0.1	**1.00**
Agent33 ◇	569.85 ± 956.24	0.07 ± 0.05	1.43 ± 0.07	0.69 ± 0.12	0.69 ± 0.12	**1.00**
Sontag ◇	1708.57 ± 2728.7	0.09 ± 0.14	1.33 ± 0.26	0.78 ± 0.14	0.78 ± 0.13	0.97
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	2805.01 ± 4277.85	0.09 ± 0.16	1.26 ± 0.28	0.83 ± 0.13	0.84 ± 0.11	0.96
ParsCat2 ⋆	2641.97 ± 4328.13	0.11 ± 0.16	1.27 ± 0.29	0.79 ± 0.13	0.80 ± 0.12	0.96
Fitness Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	**7.88 ± 2.6**	0.08 ± 0.05	1.48 ± 0.08	0.77 ± 0.09	0.77 ± 0.09	**1.00**
AgentGP ●	121.18 ± 359.01	0.08 ± 0.12	1.48 ± 0.17	0.76 ± 0.08	0.76 ± 0.06	0.99
FSEGA2019 ●	2461.25 ± 3791.92	**0.05 ± 0.04**	**1.51 ± 0.06**	0.77 ± 0.11	0.77 ± 0.11	**1.00**
AgentHerb ◇	9.57 ± 3.75	0.06 ± 0.03	1.48 ± 0.05	0.6 ± 0.09	0.6 ± 0.09	**1.00**
Agent33 ◇	4209.9 ± 6395.93	0.09 ± 0.13	1.46 ± 0.18	0.78 ± 0.11	0.79 ± 0.09	0.99
Sontag ◇	3223.29 ± 6109.35	0.06 ± 0.05	1.49 ± 0.07	0.73 ± 0.11	0.73 ± 0.11	**1.00**
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	5035.09 ± 7372.96	0.08 ± 0.04	1.47 ± 0.05	**0.8 ± 0.08**	**0.80 ± 0.08**	**1.00**
ParsCat2 ⋆	4005.65 ± 5637.52	0.07 ± 0.05	1.49 ± 0.07	0.77 ± 0.11	0.77 ± 0.11	**1.00**

Table B.5: Performance of *ANESIA-Random* - Ablation Study2 - over domain Fitness (1440 ×2 profiles = 2880 simulations)

Flight Booking Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	579.56 ± 684.55	0.34 ± 0.27	**1.16 ± 0.63**	**0.66 ± 0.21**	0.77 ± 0.24	0.46
AgentGP ●	529.11 ± 275.89	0.43 ± 0.24	0.38 ± 0.57	0.55 ± 0.12	0.65 ± 0.16	0.32
FSEGA2019 ●	3146.95 ± 4221.66	0.46 ± 0.24	0.24 ± 0.49	0.60 ± 0.2	**1.0 ± 0.0**	0.21
AgentHerb ◇	**553.35 ± 1149.49**	**0.12 ± 0.2**	**1.16 ± 0.47**	0.52 ± 0.16	0.52 ± 0.17	0.87
Agent33 ◇	1633.69 ± 2938.1	**0.12 ± 0.2**	1.01 ± 0.43	0.46 ± 0.15	0.45 ± 0.16	**0.87**
Sontag ◇	3183.22 ± 4665.68	0.38 ± 0.27	0.48 ± 0.62	**0.66 ± 0.21**	0.91 ± 0.11	0.39
AgreeableAgent ◇	3320.16 ± 4423.85	0.47 ± 0.23	0.25 ± 0.51	0.60 ± 0.19	0.98 ± 0.02	0.20
PonpokoAgent ⋆	3631.62 ± 4792.51	0.4 ± 0.26	0.39 ± 0.56	0.64 ± 0.2	0.93 ± 0.06	0.33
ParsCat2 ⋆	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00

Flight Booking Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-rand	423.39 ± 441.28	0.14 ± 0.36	1.37 ± 0.65	0.65 ± 0.27	0.71 ± 0.2	0.64
AgentGP ●	379.36 ± 186.91	0.3 ± 0.35	0.94 ± 0.65	0.55 ± 0.24	0.68 ± 0.16	0.69
FSEGA2019 ●	1785.01 ± 2478.1	0.43 ± 0.37	0.69 ± 0.66	0.59 ± 0.32	**0.90 ± 0.04**	0.52
AgentHerb ◇	**15.74 ± 40.39**	**0.06 ± 0.07**	**1.38 ± 0.14**	0.52 ± 0.18	0.52 ± 0.18	**1.00**
Agent33 ◇	844.28 ± 1399.65	0.12 ± 0.19	1.22 ± 0.34	0.47 ± 0.16	0.49 ± 0.16	0.94
Sontag ◇	1774.16 ± 2450.49	0.3 ± 0.34	0.92 ± 0.61	**0.69 ± 0.29**	0.87 ± 0.07	0.70
AgreeableAgent ◇	1959.73 ± 2285.3	0.17 ± 0.22	1.1 ± 0.37	0.64 ± 0.25	0.68 ± 0.23	0.91
PonpokoAgent ⋆	1981.05 ± 2660.34	0.37 ± 0.36	0.76 ± 0.64	0.63 ± 0.32	0.89 ± 0.07	0.59
ParsCat2 ⋆	1253.69 ± 1916.01	0.18 ± 0.21	1.17 ± 0.4	0.67 ± 0.22	0.71 ± 0.19	0.91

Table B.6: Performance of *ANESIA-Random* - Ablation Study2 - over domain Flight Booking (1440 ×2 profiles = 2880 simulations)

Itex Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	617.8 ± 561.02	0.14 ± 0.2	1.13 ± 0.44	0.67 ± 0.18	0.74 ± 0.22	0.53
AgentGP ●	378.94 ± 185.01	0.11 ± 0.17	0.6 ± 0.36	0.53 ± 0.24	0.53 ± 0.26	0.80
FSEGA2019 ●	2987.34 ± 3904.42	0.1 ± 0.15	0.68 ± 0.35	0.66 ± 0.2	0.68 ± 0.21	0.85
AgentHerb ◇	**9.19 ± 5.43**	**0.04 ± 0.06**	**1.14 ± 0.09**	0.26 ± 0.11	0.26 ± 0.11	**1.00**
Agent33 ◇	1731.26 ± 2910.4	0.06 ± 0.1	0.87 ± 0.3	0.41 ± 0.18	0.41 ± 0.18	0.96
Sontag ◇	2993.49 ± 4288.98	0.13 ± 0.18	0.65 ± 0.41	**0.69 ± 0.19**	0.75 ± 0.18	0.77
AgreeableAgent ◇	3950.25 ± 4843.71	0.16 ± 0.2	0.49 ± 0.38	0.67 ± 0.22	0.75 ± 0.23	0.69
PonpokoAgent ⋆	3704.35 ± 4514.84	0.26 ± 0.21	0.35 ± 0.4	0.66 ± 0.2	**0.86 ± 0.12**	0.46
ParsCat2 ⋆	2781.19 ± 3545.69	0.14 ± 0.18	0.62 ± 0.42	0.63 ± 0.21	0.67 ± 0.23	0.75
Itex Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	632.74 ± 580.27	0.12 ± 0.27	**1.19 ± 0.47**	0.60 ± 0.26	**0.75 ± 0.19**	0.75
AgentGP ●	490.14 ± 513.72	0.14 ± 0.24	0.9 ± 0.42	0.65 ± 0.26	0.73 ± 0.21	0.83
FSEGA2019 ●	1652.04 ± 2586.89	0.05 ± 0.13	1.1 ± 0.25	0.64 ± 0.22	0.65 ± 0.21	0.96
AgentHerb ◇	**6.5 ± 4.92**	**0.04 ± 0.05**	1.2 ± 0.08	0.36 ± 0.14	0.36 ± 0.14	**1.00**
Agent33 ◇	504.73 ± 674.86	0.05 ± 0.09	1.16 ± 0.17	0.43 ± 0.18	0.44 ± 0.18	0.99
Sontag ◇	1754.89 ± 2934.17	0.05 ± 0.13	1.12 ± 0.26	0.63 ± 0.22	0.65 ± 0.21	0.96
AgreeableAgent ◇	3252.78 ± 4395.75	0.07 ± 0.18	0.98 ± 0.31	**0.73 ± 0.24**	0.77 ± 0.2	0.92
PonpokoAgent ⋆	2817.47 ± 4135.53	0.08 ± 0.19	1.01 ± 0.34	0.72 ± 0.21	0.76 ± 0.16	0.91
ParsCat2 ⋆	1782.24 ± 2686.54	0.04 ± 0.13	1.11 ± 0.24	0.64 ± 0.23	0.66 ± 0.22	0.96

Table B.7: Performance of *ANESIA-Random* - Ablation Study2 - over domain ItexVSCypress (1440 ×2 profiles = 2880 simulations)

Outfit Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	231.72 ± 201.41	0.16 ± 0.25	1.16 ± 0.61	**0.91 ± 0.22**	**0.95 ± 0.1**	0.70
AgentGP ●	434.14 ± 336.49	0.17 ± 0.25	0.85 ± 0.64	0.78 ± 0.2	0.91 ± 0.08	0.69
FSEGA2019 ●	329.94 ± 300.72	**0.02 ± 0.1**	1.29 ± 0.35	0.81 ± 0.14	0.82 ± 0.12	0.96
AgentHerb ◇	**4.03 ± 1.69**	0.03 ± 0.08	**1.54 ± 0.17**	0.59 ± 0.19	0.59 ± 0.19	**1.00**
Agent33 ◇	133.28 ± 184.98	0.03 ± 0.1	1.45 ± 0.34	0.66 ± 0.15	0.67 ± 0.15	0.97
Sontag ◇	330.15 ± 416.61	**0.02 ± 0.11**	1.33 ± 0.37	0.79 ± 0.13	0.81 ± 0.11	0.96
AgreeableAgent ◇	606.94 ± 603.53	0.06 ± 0.17	1.04 ± 0.46	0.89 ± 0.16	0.93 ± 0.09	0.89
PonpokoAgent ⋆	515.5 ± 528.72	0.10 ± 0.19	1.04 ± 0.48	0.86 ± 0.15	0.92 ± 0.06	0.86
ParsCat2 ⋆	457.02 ± 446.94	0.03 ± 0.12	1.19 ± 0.37	0.85 ± 0.13	0.86 ± 0.11	0.95

Outfit Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-rand	244.09 ± 184.21	**0.02 ± 0.35**	1.48 ± 0.65	**0.85 ± 0.31**	**0.92 ± 0.12**	0.74
AgentGP ●	471.28 ± 350.46	0.27 ± 0.38	0.99 ± 0.70	0.70 ± 0.32	**0.92 ± 0.06**	0.68
FSEGA2019 ●	354.92 ± 351.88	0.04 ± 0.16	1.45 ± 0.34	0.8 ± 0.16	0.82 ± 0.12	0.96
AgentHerb ◇	**5.81 ± 3.33**	**0.02 ± 0.06**	**1.57 ± 0.16**	0.6 ± 0.17	0.60 ± 0.17	**1.00**
Agent33 ◇	238.07 ± 344.83	0.04 ± 0.16	1.49 ± 0.35	0.69 ± 0.17	0.71 ± 0.14	0.96
Sontag ◇	324.25 ± 378.77	0.02 ± 0.13	1.5 ± 0.28	0.8 ± 0.14	0.81 ± 0.11	0.97
AgreeableAgent ◇	669.56 ± 600.28	0.09 ± 0.25	1.25 ± 0.46	**0.85 ± 0.23**	**0.92 ± 0.12**	0.90
PonpokoAgent ⋆	536.27 ± 546.4	0.12 ± 0.27	1.27 ± 0.5	0.83 ± 0.23	0.91 ± 0.06	0.88
ParsCat2 ⋆	498.14 ± 487.94	0.05 ± 0.18	1.39 ± 0.36	0.83 ± 0.17	0.87 ± 0.1	0.95

Table B.8: Performance of *ANESIA-Random* - Ablation Study2 - over domain Outfit (1440 ×2 profiles = 2880 simulations)

Airport Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	414.88 ± 381.14	**0.1 ± 0.5**	**1.43 ± 0.71**	0.61 ± 0.16	**0.86 ± 0.17**	0.58
AgentGP ●	159.14 ± 137.0	0.08 ± 0.22	**1.43 ± 0.32**	0.64 ± 0.13	0.65 ± 0.13	0.96
FSEGA2019 ●	994.88 ± 1324.82	**0.1 ± 0.24**	1.41 ± 0.34	0.75 ± 0.17	0.75 ± 0.16	0.96
AgentHerb ◇	**9.9 ± 3.64**	0.19 ± 0.22	1.28 ± 0.24	0.5 ± 0.23	0.5 ± 0.23	**1.00**
Agent33 ◇	971.03 ± 1467.42	0.25 ± 0.31	1.16 ± 0.44	0.55 ± 0.16	0.55 ± 0.17	0.93
Sontag ◇	1101.3 ± 1448.19	0.14 ± 0.28	1.35 ± 0.41	0.72 ± 0.19	0.74 ± 0.18	0.94
AgreeableAgent ◇	1778.16 ± 2007.86	0.17 ± 0.32	1.29 ± 0.45	0.79 ± 0.2	0.81 ± 0.19	0.92
PonpokoAgent ⋆	1716.79 ± 2102.82	0.21 ± 0.35	1.24 ± 0.5	**0.81 ± 0.15**	0.86 ± 0.1	0.88
ParsCat2 ⋆	1137.39 ± 1502.72	0.14 ± 0.27	1.35 ± 0.38	0.72 ± 0.19	0.73 ± 0.19	0.95
Airport Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	328.8 ± 325.31	0.07 ± 0.24	**1.52 ± 0.61**	0.53 ± 0.26	**0.84 ± 0.14**	0.58
AgentGP ●	116.17 ± 102.35	0.07 ± 0.15	1.1 ± 0.43	0.7 ± 0.21	0.74 ± 0.17	0.92
FSEGA2019 ●	832.59 ± 857.95	0.07 ± 0.12	1.05 ± 0.37	0.72 ± 0.2	0.75 ± 0.18	0.95
AgentHerb ◇	**4.42 ± 3.69**	**0.03 ± 0.08**	1.51 ± 0.15	0.59 ± 0.13	0.59 ± 0.13	**1.00**
Agent33 ◇	255.76 ± 452.66	0.07 ± 0.09	1.34 ± 0.17	0.62 ± 0.15	0.62 ± 0.15	**1.00**
Sontag ◇	754.03 ± 784.9	0.08 ± 0.15	1.06 ± 0.41	0.72 ± 0.18	0.76 ± 0.13	0.92
AgreeableAgent ◇	1142.5 ± 1144.36	0.07 ± 0.13	0.89 ± 0.4	**0.79 ± 0.22**	0.82 ± 0.17	0.93
PonpokoAgent ⋆	1066.46 ± 999.52	0.12 ± 0.18	0.86 ± 0.44	0.75 ± 0.23	**0.84 ± 0.1**	0.85
ParsCat2 ⋆	875.01 ± 985.44	0.08 ± 0.13	1.02 ± 0.38	0.73 ± 0.18	0.76 ± 0.14	0.94

Table B.9: Performance of ANESIA-DRL - Ablation Study1 - over domain AIRPORT (1440 ×2 profiles = 2880 simulations)

Camera Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-DRL	340.58 ± 320.93	**0.04** ± **0.32**	**1.47** ± **0.62**	0.64 ± 0.14	**0.86** ± **0.11**	0.57
AgentGP ●	105.21 ± 268.21	0.17 ± 0.23	1.12 ± 0.45	0.65 ± 0.19	0.67 ± 0.19	0.88
FSEGA2019 ●	59.61 ± 38.96	0.19 ± 0.29	1.03 ± 0.55	0.76 ± 0.16	0.83 ± 0.1	0.80
AgentHerb ◇	**4.99 ± 2.94**	0.06 ± 0.08	1.41 ± 0.2	0.53 ± 0.22	0.53 ± 0.22	**1.00**
Agent33 ◇	422.69 ± 630.12	0.06 ± 0.07	1.41 ± 0.19	0.69 ± 0.18	0.69 ± 0.18	**1.00**
Sontag ◇	704.9 ± 974.92	0.07 ± 0.14	1.35 ± 0.29	0.74 ± 0.15	0.74 ± 0.15	0.97
AgreeableAgent ◇	1240.49 ± 1523.91	0.12 ± 0.23	1.1 ± 0.45	**0.80** ± **0.18**	0.84 ± 0.16	0.89
PonpokoAgent ⋆	1094.52 ± 1461.7	0.17 ± 0.28	1.06 ± 0.54	0.79 ± 0.16	**0.86** ± **0.08**	0.81
ParsCat2 ⋆	873.76 ± 1259.51	0.09 ± 0.16	1.23 ± 0.34	0.74 ± 0.18	0.76 ± 0.18	0.95
Camera Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA-DRL	280.15 ± 286.97	0.06 ± 0.2	**1.47** ± **0.57**	0.47 ± 0.37	**0.88** ± **0.13**	0.64
AgentGP ●	83.22 ± 220.98	0.14 ± 0.16	1.04 ± 0.49	0.61 ± 0.25	0.7 ± 0.11	0.88
AgentHerb ◇	**3.15 ± 0.77**	0.05 ± 0.1	1.4 ± 0.19	0.5 ± 0.19	0.5 ± 0.19	**1.00**
FSEGA2019 ●	52.5 ± 41.27	0.09 ± 0.17	1.02 ± 0.48	0.73 ± 0.3	0.85 ± 0.1	0.87
Agent33 ◇	180.65 ± 213.62	0.05 ± 0.08	1.39 ± 0.2	0.68 ± 0.18	0.68 ± 0.18	**1.00**
Sontag ◇	637.83 ± 979.8	**0.03** ± **0.08**	1.26 ± 0.29	**0.79** ± **0.17**	0.8 ± 0.13	0.98
AgreeableAgent ◇	1115.14 ± 1334.16	0.07 ± 0.14	0.9 ± 0.38	0.76 ± 0.29	0.84 ± 0.17	0.91
PonpokoAgent ⋆	821.57 ± 1069.73	0.1 ± 0.18	0.99 ± 0.52	0.74 ± 0.33	**0.88** ± **0.06**	0.83
ParsCat2 ⋆	666.97 ± 968.01	0.06 ± 0.11	1.15 ± 0.38	0.76 ± 0.21	0.8 ± 0.13	0.95

Table B.10: Performance of ANESIA-DRL - Ablation Study1 - over domain Camera (1440 ×2 profiles = 2880 simulations)

Energy Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	1397.63 ± 1143.1	0.08 ± 0.28	**1.09** ± **0.49**	0.56 ± 0.12	**0.89** ± **0.16**	0.36
AgentGP ●	131.72 ± 319.48	0.29 ± 0.28	0.66 ± 0.48	0.5 ± 0.16	0.51 ± 0.19	0.68
FSEGA2019 ●	227.86 ± 126.06	0.57 ± 0.24	0.19 ± 0.39	0.57 ± 0.14	0.84 ± 0.09	0.20
AgentHerb ◇	**22.92 ± 14.14**	**0.07** ± **0.07**	1.08 ± 0.08	0.24 ± 0.18	0.24 ± 0.18	**1.00**
Agent33 ◇	10734.09 ± 15636.62	0.22 ± 0.24	0.81 ± 0.43	0.51 ± 0.1	0.52 ± 0.12	0.80
Sontag ◇	12776.72 ± 18396.44	0.33 ± 0.31	0.57 ± 0.5	**0.65** ± **0.17**	0.76 ± 0.14	0.57
AgreeableAgent ◇	16867.8 ± 20128.27	0.41 ± 0.31	0.42 ± 0.47	0.6 ± 0.17	0.72 ± 0.2	0.46
PonpokoAgent ⋆	15871.0 ± 19512.87	0.56 ± 0.26	0.2 ± 0.4	0.58 ± 0.17	**0.89** ± **0.13**	0.21
ParsCat2 ⋆	11643.55 ± 14293.92	0.39 ± 0.3	0.49 ± 0.49	0.59 ± 0.15	0.68 ± 0.17	0.50
Energy Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	1332.93 ± 1083.96	0.09 ± 0.39	0.46 ± 0.55	**1.1** ± **0.22**	**0.86** ± **0.16**	0.41
AgentGP ●	46.98 ± 167.92	0.55 ± 0.39	0.53 ± 0.57	0.4 ± 0.19	0.56 ± 0.16	0.47
FSEGA2019 ●	225.93 ± 125.07	0.64 ± 0.39	0.36 ± 0.51	0.45 ± 0.28	0.83 ± 0.07	0.34
AgentHerb ◇	**34.68 ± 26.82**	**0.08** ± **0.07**	**1.1** ± **0.1**	0.26 ± 0.17	0.26 ± 0.17	1.00
Agent33 ◇	13342.01 ± 16011.07	0.22 ± 0.26	0.99 ± 0.38	0.43 ± 0.17	0.46 ± 0.16	0.88
Sontag ◇	17135.63 ± 22558.8	0.45 ± 0.38	0.66 ± 0.54	**0.52** ± **0.24**	0.70 ± 0.13	0.61
AgreeableAgent ◇	18105.12 ± 22793.05	0.24 ± 0.29	0.92 ± 0.39	0.45 ± 0.26	0.48 ± 0.26	0.86
PonpokoAgent ⋆	18432.89 ± 22498.15	0.59 ± 0.41	0.42 ± 0.54	0.48 ± 0.3	**0.86** ± **0.08**	0.39
ParsCat2 ⋆	14964.65 ± 18228.5	0.42 ± 0.38	0.7 ± 0.54	**0.52** ± **0.24**	0.67 ± 0.15	0.63

Table B.11: Performance of ANESIA-DRL - Ablation Study1 - over domain Energy (1440 ×2 profiles = 2880 simulations)

Grocery Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	277.41 ± 266.23	0.35 ± 0.3	**1.24** ± **0.18**	0.55 ± 0.35	**0.76** ± **0.12**	0.73
AgentGP ●	102.75 ± 249.25	0.24 ± 0.2	1.22 ± 0.41	0.68 ± 0.24	0.75 ± 0.13	0.91
FSEGA2019 ●	244.66 ± 356.63	0.25 ± 0.15	1.18 ± 0.32	0.62 ± 0.2	0.65 ± 0.13	0.95
AgentHerb ◇	**5.22 ± 2.17**	0.26 ± 0.08	1.23 ± 0.15	0.51 ± 0.15	0.51 ± 0.15	**1.00**
Agent33 ◇	90.96 ± 103.58	0.25 ± 0.08	**1.24** ± **0.15**	0.57 ± 0.17	0.57 ± 0.17	1.00
Sontag ◇	421.19 ± 754.88	0.22 ± 0.09	**1.24** ± **0.19**	0.65 ± 0.15	0.65 ± 0.13	0.99
AgreeableAgent ◇	1018.44 ± 1054.97	0.26 ± 0.16	1.0 ± 0.28	0.65 ± 0.19	0.69 ± 0.09	0.94
PonpokoAgent ⋆	562.66 ± 838.23	0.25 ± 0.17	1.14 ± 0.35	0.64 ± 0.21	0.69 ± 0.1	0.93
ParsCat2 ⋆	467.61 ± 675.5	**0.23** ± **0.11**	1.22 ± 0.24	**0.66** ± **0.17**	0.67 ± 0.13	0.97

Grocery Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	389.41 ± 336.37	0.35 ± 0.45	**1.33** ± **0.69**	0.64 ± 0.16	**0.75** ± **0.14**	0.60
AgentGP ●	164.79 ± 327.31	0.33 ± 0.33	1.2 ± 0.51	0.68 ± 0.12	0.72 ± 0.1	0.85
FSEGA2019 ●	336.73 ± 437.68	0.26 ± 0.18	1.28 ± 0.27	0.66 ± 0.11	0.67 ± 0.1	0.96
AgentHerb ◇	**8.78 ± 3.58**	**0.22** ± **0.09**	1.31 ± 0.12	0.53 ± 0.11	0.53 ± 0.11	**1.00**
Agent33 ◇	829.95 ± 1243.28	0.32 ± 0.24	1.22 ± 0.38	0.63 ± 0.09	0.64 ± 0.09	0.92
Sontag ◇	591.79 ± 969.8	0.25 ± 0.14	1.31 ± 0.21	0.64 ± 0.1	0.64 ± 0.1	0.98
AgreeableAgent ◇	1513.15 ± 1551.39	0.32 ± 0.27	1.19 ± 0.42	0.69 ± 0.11	0.71 ± 0.1	0.90
PonpokoAgent ⋆	1067.18 ± 1318.49	0.31 ± 0.29	1.2 ± 0.43	**0.72** ± **0.13**	**0.75** ± **0.11**	0.89
ParsCat2 ⋆	823.48 ± 888.27	0.28 ± 0.23	1.26 ± 0.35	0.67 ± 0.11	0.69 ± 0.1	0.93

Table B.12: Performance of ANESIA-DRL - Ablation Study1 - over domain Grocery (1440 ×2 profiles = 2880 simulations)

Fitness Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	**6.5 ± 2.6**	**0.05 ± 0.06**	**1.48 ± 0.07**	**0.87 ± 0.15**	**0.87 ± 0.15**	**1.00**
AgentGP •	507.39 ± 1234.05	0.1 ± 0.2	1.32 ± 0.37	0.67 ± 0.16	0.68 ± 0.16	0.93
FSEGA2019 •	2142.49 ± 3180.88	0.09 ± 0.14	1.32 ± 0.25	0.77 ± 0.13	0.78 ± 0.12	0.97
AgentHerb ◇	7.36 ± 3.51	**0.05 ± 0.05**	1.48 ± 0.05	0.58 ± 0.1	0.58 ± 0.1	**1.00**
Agent33 ◇	761.68 ± 1289.88	0.07 ± 0.05	1.44 ± 0.07	0.68 ± 0.12	0.68 ± 0.12	**1.00**
Sontag ◇	2813.95 ± 5044.51	0.07 ± 0.07	1.36 ± 0.14	0.78 ± 0.13	0.79 ± 0.13	**1.00**
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	3993.86 ± 5928.48	0.07 ± 0.09	1.3 ± 0.18	0.84 ± 0.11	0.84 ± 0.11	0.99
ParsCat2 ⋆	3173.43 ± 4762.17	0.08 ± 0.08	1.32 ± 0.17	0.8 ± 0.12	0.8 ± 0.12	0.99

Fitness Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	**8.05 ± 2.67**	**0.05 ± 0.05**	**1.51 ± 0.09**	**0.81 ± 0.08**	**0.81 ± 0.08**	**1.00**
AgentGP •	86.82 ± 151.37	0.06 ± 0.03	1.50 ± 0.04	0.76 ± 0.07	0.76 ± 0.07	**1.00**
FSEGA2019 •	7584.39 ± 7834.62	**0.05 ± 0.04**	**1.51 ± 0.06**	0.77 ± 0.11	0.77 ± 0.11	**1.00**
AgentHerb ◇	9.39 ± 3.46	0.06 ± 0.04	1.48 ± 0.05	0.6 ± 0.08	0.6 ± 0.08	**1.00**
Agent33 ◇	11817.2 ± 11426.17	0.08 ± 0.06	1.48 ± 0.08	0.79 ± 0.09	0.79 ± 0.09	**1.00**
Sontag ◇	8291.78 ± 8988.64	0.06 ± 0.05	1.50 ± 0.06	0.73 ± 0.11	0.73 ± 0.11	**1.00**
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	13363.2 ± 12016.88	0.08 ± 0.04	1.47 ± 0.05	**0.81 ± 0.08**	**0.81 ± 0.08**	**1.00**
ParsCat2 ⋆	11296.37 ± 10537.68	0.07 ± 0.05	1.49 ± 0.06	0.77 ± 0.1	0.77 ± 0.1	**1.00**

Table B.13: Performance of ANESIA-DRL - Ablation Study1 - over domain Fitness (1440 ×2 profiles = 2880 simulations)

Flight Booking Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	636.1 ± 658.93	0.13 ± 0.25	**1.36** ± **0.58**	0.59 ± 0.18	**0.98** ± **0.2**	0.29
AgentGP ●	505.74 ± 291.15	0.41 ± 0.24	0.47 ± 0.61	0.55 ± 0.12	0.62 ± 0.16	0.39
FSEGA2019 ●	3733.36 ± 5491.65	0.47 ± 0.23	0.23 ± 0.48	0.6 ± 0.2	1.0 ± 0.0	0.20
AgentHerb ◇	**451.6** ± **1069.53**	**0.12** ± **0.19**	1.18 ± 0.44	0.51 ± 0.16	0.51 ± 0.17	**0.89**
Agent33 ◇	1446.79 ± 3331.37	0.17 ± 0.23	0.98 ± 0.52	0.51 ± 0.17	0.51 ± 0.19	0.81
Sontag ◇	2994.05 ± 5633.23	0.36 ± 0.27	0.53 ± 0.65	0.67 ± 0.22	0.93 ± 0.09	0.41
AgreeableAgent ◇	3674.2 ± 5496.21	0.46 ± 0.24	0.26 ± 0.51	0.6 ± 0.2	**0.98** ± **0.02**	0.21
PonpokoAgent ⋆	3749.03 ± 6108.73	0.42 ± 0.25	0.37 ± 0.57	0.63 ± 0.2	0.93 ± 0.04	0.30
ParsCat2 ⋆	1887.62 ± 3405.91	0.3 ± 0.26	0.76 ± 0.65	0.6 ± 0.18	0.67 ± 0.21	0.59

Flight Booking Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	572.81 ± 538.65	**0.06** ± **0.36**	**1.45** ± **0.64**	0.44 ± 0.29	**0.94** ± **0.17**	0.33
AgentGP ●	416.07 ± 215.14	0.35 ± 0.36	0.84 ± 0.68	0.52 ± 0.25	0.69 ± 0.17	0.62
FSEGA2019 ●	2539.12 ± 4104.76	0.46 ± 0.37	0.63 ± 0.66	0.56 ± 0.33	0.9 ± 0.04	0.48
AgentHerb ◇	**24.15** ± **115.88**	**0.06** ± **0.1**	1.37 ± 0.2	0.52 ± 0.19	0.52 ± 0.19	**0.99**
Agent33 ◇	1111.08 ± 2227.52	0.13 ± 0.21	1.2 ± 0.38	0.46 ± 0.16	0.48 ± 0.16	0.92
Sontag ◇	2525.82 ± 4221.99	0.34 ± 0.35	0.84 ± 0.64	**0.65** ± **0.3**	0.87 ± 0.08	0.65
AgreeableAgent ◇	3026.83 ± 4474.59	0.24 ± 0.29	0.98 ± 0.49	0.59 ± 0.27	0.67 ± 0.24	0.81
PonpokoAgent ⋆	2600.7 ± 4136.52	0.4 ± 0.37	0.72 ± 0.65	0.61 ± 0.33	0.9 ± 0.07	0.55
ParsCat2 ⋆	1577.21 ± 2510.69	0.23 ± 0.27	1.07 ± 0.5	0.63 ± 0.24	0.7 ± 0.19	0.84

Table B.14: Performance of ANESIA-DRL - Ablation Study1 - over domain Flight Booking (1440 ×2 profiles = 2880 simulations)

Itex Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	555.81 ± 532.81	**0.04** ± **0.16**	**1.27** ± **0.46**	0.56 ± 0.12	**0.91** ± **0.16**	0.27
AgentGP ●	371.01 ± 177.31	0.12 ± 0.18	0.58 ± 0.38	0.51 ± 0.23	0.52 ± 0.27	0.77
FSEGA2019 ●	2714.98 ± 4328.78	0.11 ± 0.16	0.67 ± 0.37	0.66 ± 0.21	0.69 ± 0.21	0.82
AgentHerb ◇	**9.12** ± **6.25**	0.05 ± 0.06	1.15 ± 0.08	0.26 ± 0.11	0.26 ± 0.11	**1.00**
Agent33 ◇	1850.84 ± 3677.84	0.07 ± 0.12	0.85 ± 0.34	0.42 ± 0.18	0.41 ± 0.18	0.92
Sontag ◇	2965.81 ± 5008.5	0.14 ± 0.19	0.63 ± 0.43	**0.69** ± **0.19**	0.76 ± 0.17	0.73
AgreeableAgent ◇	3933.17 ± 5861.42	0.15 ± 0.19	0.49 ± 0.37	0.67 ± 0.22	0.75 ± 0.23	0.70
PonpokoAgent ⋆	3909.99 ± 5863.55	0.27 ± 0.2	0.32 ± 0.4	0.65 ± 0.19	0.85 ± 0.14	0.43
ParsCat2 ⋆	2808.13 ± 4315.82	0.15 ± 0.19	0.61 ± 0.43	0.62 ± 0.21	0.67 ± 0.24	0.72

Itex Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	381.27 ± 275.45	**0.04** ± **0.02**	**1.24** ± **0.54**	0.54 ± 0.3	**0.82** ± **0.15**	0.51
AgentGP ●	542.41 ± 515.26	0.24 ± 0.29	0.73 ± 0.5	0.56 ± 0.27	0.7 ± 0.22	0.69
FSEGA2019 ●	806.26 ± 1067.19	0.08 ± 0.19	1.03 ± 0.34	0.6 ± 0.24	0.64 ± 0.22	0.91
AgentHerb ◇	**6.44** ± **5.1**	**0.04** ± **0.05**	1.21 ± 0.07	0.37 ± 0.14	0.37 ± 0.14	**1.00**
Agent33 ◇	289.54 ± 402.6	0.05 ± 0.09	1.16 ± 0.18	0.44 ± 0.18	0.44 ± 0.18	0.98
Sontag ◇	881.04 ± 1273.89	0.07 ± 0.19	1.06 ± 0.36	0.61 ± 0.24	0.65 ± 0.22	0.91
AgreeableAgent ◇	1398.88 ± 1655.48	0.11 ± 0.22	0.92 ± 0.37	**0.69** ± **0.27**	0.76 ± 0.22	0.87
PonpokoAgent ⋆	1190.43 ± 1689.85	0.13 ± 0.24	0.91 ± 0.43	0.68 ± 0.24	0.77 ± 0.16	0.83
ParsCat2 ⋆	907.87 ± 1153.59	0.08 ± 0.2	1.03 ± 0.36	0.61 ± 0.25	0.65 ± 0.23	0.90

Table B.15: Performance of ANESIA-DRL - Ablation Study1 - over domain ItexVS-cypress (1440 ×2 profiles = 2880 simulations)

Outfit Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	355.37 ± 256.24	**0.05** ± **0.27**	**1.57** ± **0.67**	0.76 ± 0.25	**0.98** ± **0.06**	0.54
AgentGP ●	704.03 ± 524.83	0.27 ± 0.27	0.71 ± 0.72	0.71 ± 0.22	0.91 ± 0.09	0.51
FSEGA2019 ●	690.15 ± 766.83	0.04 ± 0.14	1.27 ± 0.42	0.8 ± 0.15	0.83 ± 0.12	0.93
AgentHerb ◇	**4.04 ± 1.77**	**0.03** ± **0.07**	1.55 ± 0.17	0.59 ± 0.18	0.59 ± 0.18	**1.00**
Agent33 ◇	244.91 ± 419.1	0.03 ± 0.1	1.47 ± 0.34	0.67 ± 0.14	0.67 ± 0.14	0.96
Sontag ◇	600.69 ± 765.69	0.04 ± 0.14	1.32 ± 0.42	0.79 ± 0.14	0.81 ± 0.11	0.93
AgreeableAgent ◇	1207.69 ± 1217.1	0.1 ± 0.21	0.98 ± 0.53	**0.86** ± **0.19**	0.94 ± 0.08	0.82
PonpokoAgent ⋆	989.18 ± 1035.21	0.13 ± 0.21	1.01 ± 0.55	0.84 ± 0.17	0.92 ± 0.06	0.80
ParsCat2 ⋆	807.16 ± 846.0	0.06 ± 0.16	1.15 ± 0.45	0.83 ± 0.15	0.87 ± 0.11	0.90

Outfit Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA-DRL	409.13 ± 289.53	0.05 ± 0.4	**1.63** ± **0.73**	0.67 ± 0.37	**0.98** ± **0.08**	0.57
AgentGP ●	701.88 ± 540.79	0.4 ± 0.4	0.78 ± 0.77	0.6 ± 0.34	0.93 ± 0.08	0.51
FSEGA2019 ●	632.52 ± 701.28	0.06 ± 0.22	1.41 ± 0.43	0.78 ± 0.19	0.82 ± 0.12	0.92
AgentHerb ◇	**5.8 ± 3.34**	**0.01** ± **0.05**	1.57 ± 0.16	0.6 ± 0.17	0.6 ± 0.17	**1.00**
Agent33 ◇	444.49 ± 712.8	0.05 ± 0.19	1.46 ± 0.4	0.67 ± 0.17	0.69 ± 0.14	0.94
Sontag ◇	643.55 ± 752.71	0.05 ± 0.2	1.45 ± 0.4	0.77 ± 0.18	0.81 ± 0.12	0.94
AgreeableAgent ◇	1353.41 ± 1342.23	0.12 ± 0.28	1.19 ± 0.52	**0.83** ± **0.26**	0.92 ± 0.13	0.86
PonpokoAgent ⋆	1104.35 ± 1136.18	0.17 ± 0.32	1.19 ± 0.6	0.79 ± 0.27	0.92 ± 0.06	0.81
ParsCat2 ⋆	978.63 ± 945.08	0.08 ± 0.24	1.34 ± 0.46	0.81 ± 0.21	0.86 ± 0.11	0.90

Table B.16: Performance of ANESIA-DRL - Ablation Study1 - over domain Outfit (1440 ×2 profiles = 2880 simulations)

Airport Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA	502.96 ± 350.71	0.06 ± 0.53	**1.76** ± **0.79**	0.69 ± 0.21	**0.92** ± **0.11**	0.48
AgentGP ●	774.69 ± 562.07	0.59 ± 0.53	0.72 ± 0.79	0.69 ± 0.21	0.92 ± 0.06	0.45
FSEGA2019 ●	1023.41 ± 1450.49	0.07 ± 0.26	1.49 ± 0.38	0.77 ± 0.14	0.78 ± 0.13	0.94
AgentHerb ◇	**10.58 ± 3.59**	**0.0** ± **0.01**	1.55 ± 0.11	0.59 ± 0.14	0.59 ± 0.14	1.00
Agent33 ◇	1556.13 ± 2104.58	0.19 ± 0.41	1.3 ± 0.6	0.72 ± 0.13	0.76 ± 0.09	0.82
Sontag ◇	1285.67 ± 1931.73	0.08 ± 0.27	1.47 ± 0.41	0.73 ± 0.12	0.75 ± 0.11	0.93
AgreeableAgent ◇	2206.13 ± 2724.1	0.18 ± 0.4	1.32 ± 0.59	**0.82** ± **0.18**	0.88 ± 0.12	0.84
PonpokoAgent ⋆	1988.18 ± 2717.28	0.19 ± 0.41	1.29 ± 0.6	**0.82** ± **0.16**	0.88 ± 0.07	0.82
ParsCat2 ⋆	1490.62 ± 1833.53	0.09 ± 0.3	1.44 ± 0.44	0.77 ± 0.14	0.79 ± 0.12	**0.92**

Airport Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA	387.84 ± 297.52	**0.06** ± **0.27**	**1.64** ± **0.04**	0.62 ± 0.36	**0.96** ± **0.06**	0.52
AgentGP ●	646.55 ± 497.55	0.25 ± 0.27	0.69 ± 0.68	0.61 ± 0.34	0.92 ± 0.06	0.54
FSEGA2019 ●	814.29 ± 813.53	0.05 ± 0.15	1.13 ± 0.4	0.74 ± 0.19	0.79 ± 0.13	0.92
AgentHerb ◇	**3.13 ± 1.23**	0.01 ± 0.02	1.54 ± 0.11	0.56 ± 0.12	0.56 ± 0.12	**1.00**
Agent33 ◇	182.67 ± 151.37	0.0 ± 0.01	1.46 ± 0.12	0.63 ± 0.13	0.63 ± 0.13	1.00
Sontag ◇	712.1 ± 746.85	0.06 ± 0.17	1.15 ± 0.45	0.72 ± 0.2	0.78 ± 0.12	0.89
AgreeableAgent ◇	1192.59 ± 1179.95	0.08 ± 0.19	0.93 ± 0.47	**0.79** ± **0.27**	0.88 ± 0.16	0.86
PonpokoAgent ⋆	1091.71 ± 1197.91	0.12 ± 0.22	0.91 ± 0.54	0.75 ± 0.27	0.89 ± 0.07	0.78
ParsCat2 ⋆	925.84 ± 884.29	0.06 ± 0.17	1.04 ± 0.44	0.76 ± 0.22	0.82 ± 0.13	0.89

Table B.17: Performance of fully-fledged *ANESIA* in the domain AIRPORT (1440 ×2 profiles = 2880 simulations)

Camera Domain B = 5% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA	387.64 ± 264.96	0.02 ± 0.37	**1.78 ± 0.66**	0.74 ± 0.21	**0.91 ± 0.08**	0.59
AgentGP ●	615.41 ± 511.43	0.36 ± 0.37	0.78 ± 0.73	0.7 ± 0.2	0.87 ± 0.11	0.54
FSEGA2019 ●	58.92 ± 33.98	0.09 ± 0.23	1.28 ± 0.45	0.82 ± 0.12	0.86 ± 0.05	0.89
AgentHerb ◇	**7.48 ± 3.18**	**0.01 ± 0.02**	1.43 ± 0.13	0.47 ± 0.15	0.47 ± 0.15	**1.00**
Agent33 ◇	589.89 ± 802.57	0.01 ± 0.08	1.46 ± 0.21	0.71 ± 0.14	0.71 ± 0.14	0.99
Sontag ◇	779.55 ± 1056.75	0.06 ± 0.2	1.37 ± 0.41	0.79 ± 0.12	0.81 ± 0.09	0.92
AgreeableAgent ◇	1542.18 ± 1587.82	0.11 ± 0.26	1.11 ± 0.46	**0.84 ± 0.16**	0.90 ± 0.09	0.86
PonpokoAgent ⋆	1064.8 ± 1316.41	0.16 ± 0.3	1.11 ± 0.56	0.82 ± 0.17	0.90 ± 0.06	0.80
ParsCat2 ⋆	879.17 ± 1171.17	0.1 ± 0.25	1.25 ± 0.5	0.79 ± 0.14	0.84 ± 0.09	0.87

Camera Domain B = 10% of Ω						
Agent	R_{avg}(↓)	P_{avg}(↓)	U_{soc}(↑)	U_{ind}^{total}(↑)	U_{ind}^{s}(↑)	$S_{\%}$(↑)
ANESIA	370.23 ± 256.09	0.02 ± 0.25	**1.64 ± 0.57**	0.54 ± 0.46	**0.93 ± 0.08**	0.58
AgentGP ●	555.57 ± 485.57	0.23 ± 0.24	0.72 ± 0.67	0.5 ± 0.44	0.89 ± 0.09	0.56
FSEGA2019 ●	58.76 ± 35.67	0.06 ± 0.15	1.06 ± 0.41	**0.78 ± 0.27**	0.86 ± 0.06	0.90
AgentHerb ◇	**4.63 ± 2.37**	**0.01 ± 0.02**	1.41 ± 0.13	0.45 ± 0.13	0.45 ± 0.13	**1.00**
Agent33 ◇	272.72 ± 306.3	0.01 ± 0.05	1.4 ± 0.22	0.64 ± 0.21	0.65 ± 0.2	0.99
Sontag ◇	731.21 ± 956.39	0.03 ± 0.12	1.24 ± 0.38	0.77 ± 0.22	0.82 ± 0.1	0.94
AgreeableAgent ◇	1231.69 ± 1291.81	0.07 ± 0.17	0.87 ± 0.4	0.76 ± 0.33	0.88 ± 0.15	0.86
PonpokoAgent ⋆	898.9 ± 1035.82	0.12 ± 0.21	0.9 ± 0.54	0.69 ± 0.39	0.90 ± 0.06	0.77
ParsCat2 ⋆	721.07 ± 836.11	0.05 ± 0.15	1.11 ± 0.44	0.75 ± 0.27	0.84 ± 0.09	0.90

Table B.18: Performance of fully-fledged *ANESIA* in the domain Camera (1440 ×2 profiles = 2880 simulations)

Energy Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	1885.12 ± 1220.17	0.59 ± 0.24	**1.15** ± **0.36**	0.57 ± 0.17	**0.97** ± **0.04**	0.15
AgentGP ●	776.26 ± 1337.29	0.36 ± 0.34	0.58 ± 0.59	**0.71** ± **0.21**	0.92 ± 0.02	0.49
FSEGA2019 ●	227.55 ± 94.34	0.53 ± 0.29	0.26 ± 0.46	0.6 ± 0.18	0.91 ± 0.06	0.24
AgentHerb ◇	**209.5 ± 390.96**	**0.01** ± **0.03**	1.11 ± 0.13	0.21 ± 0.15	0.21 ± 0.15	**1.00**
Agent33 ◇	9736.06 ± 11596.61	0.11 ± 0.24	0.98 ± 0.42	0.5 ± 0.16	0.5 ± 0.17	0.86
Sontag ◇	16164.98 ± 19874.32	0.3 ± 0.33	0.65 ± 0.56	0.66 ± 0.17	0.78 ± 0.12	0.59
AgreeableAgent ◇	20110.81 ± 19539.26	0.21 ± 0.31	0.69 ± 0.45	0.69 ± 0.19	0.77 ± 0.18	0.71
PonpokoAgent ⋆	18600.12 ± 19064.62	0.33 ± 0.34	0.54 ± 0.52	0.69 ± 0.2	0.87 ± 0.1	0.52
ParsCat2 ⋆	13441.85 ± 13936.75	0.25 ± 0.32	0.72 ± 0.52	0.64 ± 0.18	0.71 ± 0.18	0.65
Energy Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	2207.35 ± 1420.7	0.74 ± 0.36	**1.21** ± **0.44**	0.39 ± 0.28	**0.96** ± **0.05**	0.19
AgentGP ●	**157.6 ± 278.3**	0.07 ± 0.21	1.13 ± 0.28	**0.60** ± **0.31**	0.62 ± 0.3	0.94
FSEGA2019 ●	235.12 ± 104.56	0.48 ± 0.45	0.59 ± 0.6	0.56 ± 0.32	0.89 ± 0.06	0.49
AgentHerb ◇	200.51 ± 378.53	**0.01** ± **0.03**	1.15 ± 0.08	0.21 ± 0.15	0.21 ± 0.15	**1.00**
Agent33 ◇	24080.45 ± 23658.48	0.14 ± 0.3	1.14 ± 0.44	0.53 ± 0.2	0.57 ± 0.18	0.87
Sontag ◇	33677.34 ± 37660.47	0.4 ± 0.43	0.73 ± 0.62	0.54 ± 0.26	0.74 ± 0.14	0.59
AgreeableAgent ◇	35654.55 ± 36140.39	0.15 ± 0.29	1.04 ± 0.41	0.57 ± 0.3	0.62 ± 0.29	0.88
PonpokoAgent ⋆	37176.59 ± 37296.11	0.47 ± 0.45	0.61 ± 0.61	0.57 ± 0.32	0.88 ± 0.08	0.50
ParsCat2 ⋆	24819.35 ± 24326.69	0.35 ± 0.44	0.82 ± 0.64	0.52 ± 0.24	0.68 ± 0.16	0.63

Table B.19: Performance of fully-fledged *ANESIA* in the domain Energy (1440 ×2 profiles = 2880 simulations)

Grocery Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	358.62 ± 240.22	0.26 ± 0.37	**1.57 ± 0.65**	0.65 ± 0.43	**0.93 ± 0.06**	0.70
AgentGP ●	507.2 ± 380.16	0.27 ± 0.37	1.0 ± 0.68	0.62 ± 0.42	0.90 ± 0.09	0.69
FSEGA2019 ●	379.88 ± 415.61	0.01 ± 0.07	1.49 ± 0.17	**0.84 ± 0.13**	0.84 ± 0.11	0.99
AgentHerb ◇	**5.54 ± 1.73**	0.02 ± 0.05	1.53 ± 0.1	0.56 ± 0.12	0.56 ± 0.12	**1.00**
Agent33 ◇	300.19 ± 371.5	0.03 ± 0.1	1.51 ± 0.22	0.67 ± 0.16	0.68 ± 0.14	0.99
Sontag ◇	606.43 ± 837.78	**0.01 ± 0.01**	1.53 ± 0.12	0.82 ± 0.11	0.82 ± 0.11	**1.00**
AgreeableAgent ◇	1195.86 ± 1127.9	0.1 ± 0.26	1.2 ± 0.43	0.82 ± 0.31	0.92 ± 0.12	0.89
PonpokoAgent ⋆	776.33 ± 950.82	0.14 ± 0.3	1.24 ± 0.56	0.77 ± 0.34	0.92 ± 0.05	0.84
ParsCat2 ⋆	665.25 ± 700.86	0.07 ± 0.22	1.38 ± 0.42	0.79 ± 0.25	0.86 ± 0.1	0.92

Grocery Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	423.55 ± 291.29	0.4 ± 0.52	**1.65 ± 0.78**	0.76 ± 0.21	**0.91 ± 0.08**	0.65
AgentGP ●	606.72 ± 470.98	0.49 ± 0.53	0.93 ± 0.81	0.73 ± 0.21	**0.91 ± 0.08**	0.57
FSEGA2019 ●	463.94 ± 485.36	0.02 ± 0.13	1.64 ± 0.2	0.81 ± 0.11	0.81 ± 0.11	0.99
AgentHerb ◇	**7.81 ± 2.34**	**0.01 ± 0.04**	1.54 ± 0.09	0.57 ± 0.12	0.57 ± 0.12	**1.00**
Agent33 ◇	975.11 ± 984.61	0.21 ± 0.41	1.37 ± 0.63	0.78 ± 0.16	0.84 ± 0.11	0.83
Sontag ◇	822.18 ± 1146.13	**0.01 ± 0.06**	**1.65 ± 0.13**	0.8 ± 0.11	0.8 ± 0.1	**1.00**
AgreeableAgent ◇	1518.27 ± 1454.96	0.12 ± 0.34	1.46 ± 0.51	**0.87 ± 0.17**	**0.91 ± 0.11**	0.89
PonpokoAgent ⋆	1103.99 ± 1269.89	0.17 ± 0.39	1.43 ± 0.6	0.85 ± 0.15	**0.91 ± 0.06**	0.85
ParsCat2 ⋆	826.77 ± 846.74	0.08 ± 0.28	1.57 ± 0.44	0.81 ± 0.13	0.84 ± 0.1	0.93

Table B.20: Performance of fully-fledged *ANESIA* in the domain Grocery (1440 ×2 profiles = 2880 simulations)

Fitness Domain B = 5% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	8.13 ± 1.19	**0.0 ± 0.0**	**1.51 ± 0.01**	**1.0 ± 0.0**	**1.0 ± 0.0**	**1.00**
AgentGP ●	5025.1 ± 4975.25	0.01 ± 0.01	1.49 ± 0.05	0.92 ± 0.02	0.92 ± 0.02	**1.00**
FSEGA2019 ●	6462.11 ± 5295.59	**0.0 ± 0.01**	1.45 ± 0.07	0.81 ± 0.11	0.81 ± 0.11	**1.00**
AgentHerb ◇	**7.25 ± 2.01**	0.01 ± 0.02	**1.51 ± 0.04**	0.55 ± 0.07	0.55 ± 0.07	**1.00**
Agent33 ◇	3483.88 ± 2350.29	0.01 ± 0.02	1.5 ± 0.04	0.7 ± 0.09	0.7 ± 0.09	**1.00**
Sontag ◇	6117.57 ± 6137.66	**0.0 ± 0.01**	1.47 ± 0.07	0.8 ± 0.11	0.8 ± 0.11	**1.00**
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	9504.37 ± 8067.99	**0.0 ± 0.01**	1.4 ± 0.1	0.92 ± 0.05	0.92 ± 0.05	**1.00**
ParsCat2 ⋆	8187.34 ± 6755.87	0.01 ± 0.02	1.42 ± 0.09	0.87 ± 0.1	0.87 ± 0.1	**1.00**

Fitness Domain B = 10% of Ω

Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	**10.33 ± 1.05**	**0.0 ± 0.0**	**1.57 ± 0.01**	**1.0 ± 0.0**	**1.0 ± 0.0**	**1.00**
AgentGP ●	7520.09 ± 8454.27	0.01 ± 0.01	**1.57 ± 0.04**	0.93 ± 0.03	0.93 ± 0.03	**1.00**
FSEGA2019 ●	10477.08 ± 7392.38	0.01 ± 0.04	1.56 ± 0.07	0.77 ± 0.12	0.77 ± 0.12	**1.00**
AgentHerb ◇	12.52 ± 3.01	0.01 ± 0.02	1.52 ± 0.04	0.54 ± 0.07	0.54 ± 0.07	**1.00**
Agent33 ◇	18549.4 ± 14784.47	0.03 ± 0.04	1.55 ± 0.07	0.84 ± 0.09	0.84 ± 0.09	**1.00**
Sontag ◇	12031.32 ± 10180.53	0.01 ± 0.01	**1.57 ± 0.05**	0.76 ± 0.12	0.76 ± 0.12	**1.00**
AgreeableAgent ◇	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.00
PonpokoAgent ⋆	19677.92 ± 15053.14	0.01 ± 0.02	**1.57 ± 0.04**	0.9 ± 0.06	0.9 ± 0.06	**1.00**
ParsCat2 ⋆	16583.31 ± 12421.79	0.01 ± 0.02	**1.57 ± 0.05**	0.84 ± 0.1	0.84 ± 0.1	**1.00**

Table B.21: Performance of fully-fledged *ANESIA* in the domain Fitness (1440 ×2 profiles = 2880 simulations)

Flight Booking Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U^{total}_{ind}(\uparrow)$	$U^{s}_{ind}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	457.11 ± 307.19	0.04 ± 0.28	**1.54** ± **0.63**	0.66 ± 0.21	**0.9** ± **0.16**	0.43
AgentGP ●	664.13 ± 477.03	0.35 ± 0.28	0.56 ± 0.68	0.66 ± 0.2	**0.9** ± **0.08**	0.41
FSEGA2019 ●	1021.27 ± 1249.01	0.09 ± 0.2	1.14 ± 0.46	0.78 ± 0.15	0.82 ± 0.12	0.87
AgentHerb ◇	**7.7 ± 3.9**	**0.01** ± **0.05**	1.37 ± 0.09	0.42 ± 0.13	0.42 ± 0.13	**1.00**
Agent33 ◇	543.69 ± 758.12	0.05 ± 0.13	1.26 ± 0.34	0.52 ± 0.11	0.52 ± 0.11	0.95
Sontag ◇	1041.66 ± 1365.84	0.1 ± 0.21	1.13 ± 0.51	0.75 ± 0.15	0.8 ± 0.11	0.84
AgreeableAgent ◇	1838.85 ± 1648.32	0.12 ± 0.22	0.98 ± 0.47	**0.79** ± **0.17**	0.85 ± 0.12	0.82
PonpokoAgent ⋆	1500.68 ± 1771.53	0.12 ± 0.22	1.05 ± 0.51	0.81 ± 0.16	0.88 ± 0.06	0.82
ParsCat2 ⋆	1143.01 ± 1320.89	0.09 ± 0.2	1.12 ± 0.45	0.78 ± 0.17	0.82 ± 0.14	0.86
Flight Booking Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U^{total}_{ind}(\uparrow)$	$U^{s}_{ind}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	479.79 ± 300.63	0.02 ± 0.4	**1.64** ± **0.66**	0.56 ± 0.34	**0.89** ± **0.15**	0.49
AgentGP ●	698.1 ± 501.62	0.47 ± 0.39	0.61 ± 0.7	0.53 ± 0.33	**0.89** ± **0.1**	0.43
FSEGA2019 ●	1099.44 ± 1361.39	0.12 ± 0.27	1.23 ± 0.48	0.75 ± 0.22	0.83 ± 0.12	0.87
AgentHerb ◇	**9.17 ± 2.93**	**0.01** ± **0.03**	1.38 ± 0.09	0.43 ± 0.15	0.43 ± 0.15	**1.00**
Agent33 ◇	741.04 ± 1130.56	0.04 ± 0.12	1.35 ± 0.24	0.51 ± 0.15	0.51 ± 0.14	0.98
Sontag ◇	1103.64 ± 1566.31	0.12 ± 0.28	1.24 ± 0.52	0.72 ± 0.22	0.8 ± 0.11	0.86
AgreeableAgent ◇	1981.51 ± 1924.17	0.11 ± 0.26	1.17 ± 0.44	0.75 ± 0.25	0.82 ± 0.19	0.88
PonpokoAgent ⋆	1565.29 ± 1663.12	0.15 ± 0.3	1.16 ± 0.54	**0.78** ± **0.24**	0.88 ± 0.06	0.83
ParsCat2 ⋆	1135.35 ± 1244.4	0.12 ± 0.28	1.21 ± 0.5	0.74 ± 0.24	0.83 ± 0.14	0.85

Table B.22: Performance of fully-fledged *ANESIA* in the domain Flight Booking (1440 ×2 profiles = 2880 simulations)

Itex Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	564.37 ± 402.32	0.39 ± 0.15	**1.26 ± 0.09**	0.56 ± 0.17	**1.0 ± 0.0**	0.13
AgentGP ●	686.32 ± 481.19	0.32 ± 0.2	0.31 ± 0.5	0.6 ± 0.18	0.85 ± 0.14	0.29
FSEGA2019 ●	1833.21 ± 2581.66	0.14 ± 0.21	0.64 ± 0.46	0.67 ± 0.18	0.75 ± 0.17	0.68
AgentHerb ◇	**9.1 ± 2.82**	**0.0 ± 0.02**	1.19 ± 0.04	0.23 ± 0.08	0.23 ± 0.08	1.00
Agent33 ◇	1187.91 ± 2093.66	0.11 ± 0.19	0.8 ± 0.49	0.47 ± 0.16	0.46 ± 0.18	0.76
Sontag ◇	1934.54 ± 2764.67	0.14 ± 0.2	0.65 ± 0.47	**0.70 ± 0.17**	0.79 ± 0.13	0.69
AgreeableAgent ◇	2839.27 ± 3542.99	0.15 ± 0.2	0.52 ± 0.4	0.67 ± 0.17	0.76 ± 0.15	0.68
PonpokoAgent ⋆	2801.4 ± 3525.25	0.32 ± 0.2	0.28 ± 0.45	0.61 ± 0.17	0.86 ± 0.1	0.29
ParsCat2 ⋆	2080.09 ± 3082.12	0.14 ± 0.2	0.64 ± 0.46	0.68 ± 0.18	0.77 ± 0.16	0.70

Itex Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	591.32 ± 412.93	**0.06 ± 0.24**	**1.17 ± 0.41**	0.36 ± 0.27	**0.99 ± 0.03**	0.15
AgentGP ●	745.37 ± 490.4	0.46 ± 0.3	0.35 ± 0.53	0.43 ± 0.28	0.83 ± 0.16	0.32
FSEGA2019 ●	1796.16 ± 2823.98	0.2 ± 0.3	0.79 ± 0.52	0.59 ± 0.26	0.74 ± 0.16	0.70
AgentHerb ◇	**12.5 ± 4.59**	**0.0 ± 0.03**	1.2 ± 0.05	0.26 ± 0.16	0.26 ± 0.16	**1.00**
Agent33 ◇	1124.77 ± 1744.99	0.12 ± 0.25	0.95 ± 0.45	0.4 ± 0.16	0.43 ± 0.16	0.83
Sontag ◇	2100.02 ± 2721.24	0.21 ± 0.29	0.78 ± 0.51	**0.62 ± 0.26**	0.77 ± 0.12	0.71
AgreeableAgent ◇	2765.74 ± 3225.78	0.14 ± 0.25	0.81 ± 0.39	0.59 ± 0.28	0.67 ± 0.25	0.82
PonpokoAgent ⋆	2830.23 ± 3765.43	0.36 ± 0.33	0.48 ± 0.53	0.54 ± 0.32	0.88 ± 0.08	0.46
ParsCat2 ⋆	2363.75 ± 3460.21	0.21 ± 0.3	0.76 ± 0.51	0.61 ± 0.27	0.77 ± 0.16	0.69

Table B.23: Performance of fully-fledged *ANESIA* in the domain ItexVSCypress (1440 ×2 profiles = 2880 simulations)

Outfit Domain B = 5% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	516.7 ± 352.5	0.05 ± 0.4	**1.57** ± **0.73**	0.67 ± 0.37	**0.99** ± **0.02**	0.56
AgentGP ●	786.93 ± 580.41	0.39 ± 0.4	0.83 ± 0.79	0.61 ± 0.34	0.93 ± 0.07	0.53
FSEGA2019 ●	854.18 ± 890.84	0.05 ± 0.19	1.44 ± 0.39	0.79 ± 0.18	0.83 ± 0.11	0.94
AgentHerb ◇	**5.73 ± 3.37**	**0.01** ± **0.04**	**1.57** ± **0.16**	0.60 ± 0.17	0.6 ± 0.17	**1.00**
Agent33 ◇	499.38 ± 763.55	0.04 ± 0.16	1.48 ± 0.37	0.68 ± 0.17	0.70 ± 0.15	0.96
Sontag ◇	749.45 ± 868.18	0.05 ± 0.18	1.47 ± 0.38	0.78 ± 0.17	0.81 ± 0.12	0.94
AgreeableAgent ◇	1644.39 ± 1461.16	0.10 ± 0.27	1.22 ± 0.5	**0.84** ± **0.25**	0.92 ± 0.13	0.87
PonpokoAgent ⋆	1277.88 ± 1259.92	0.17 ± 0.32	1.19 ± 0.6	0.79 ± 0.27	0.92 ± 0.06	0.80
ParsCat2 ⋆	1191.12 ± 1206.94	0.08 ± 0.24	1.33 ± 0.46	0.81 ± 0.21	0.87 ± 0.11	0.90
Outfit Domain B = 10% of Ω						
Agent	$R_{avg}(\downarrow)$	$P_{avg}(\downarrow)$	$U_{soc}(\uparrow)$	$U_{ind}^{total}(\uparrow)$	$U_{ind}^{s}(\uparrow)$	$S_{\%}(\uparrow)$
ANESIA	868.7 ± 799.74	**0.02** ± **0.27**	**1.54** ± **0.67**	0.78 ± 0.25	**0.99** ± **0.04**	0.56
AgentGP ●	1105.14 ± 1567.54	0.25 ± 0.26	0.83 ± 0.75	0.75 ± 0.22	0.93 ± 0.03	0.57
FSEGA2019 ●	2026.02 ± 3050.22	0.04 ± 0.14	1.27 ± 0.41	0.81 ± 0.14	0.83 ± 0.12	0.93
AgentHerb ◇	**3.64 ± 1.91**	0.03 ± 0.08	1.54 ± 0.18	0.57 ± 0.17	0.57 ± 0.17	**1.00**
Agent33 ◇	888.24 ± 1927.38	**0.02** ± **0.09**	1.48 ± 0.31	0.68 ± 0.15	0.68 ± 0.15	0.97
Sontag ◇	2213.92 ± 3866.96	0.03 ± 0.12	1.35 ± 0.4	0.80 ± 0.13	0.81 ± 0.11	0.94
AgreeableAgent ◇	4341.84 ± 6618.34	0.08 ± 0.2	1.01 ± 0.51	**0.88** ± **0.18**	0.95 ± 0.08	0.85
PonpokoAgent ⋆	3897.28 ± 5823.6	0.12 ± 0.2	1.04 ± 0.52	0.85 ± 0.17	0.92 ± 0.06	0.83
ParsCat2 ⋆	3089.0 ± 4560.45	0.05 ± 0.15	1.18 ± 0.43	0.84 ± 0.14	0.87 ± 0.1	0.92

Table B.24: Performance of fully-fledged *ANESIA* in the domain Outfit (1440 ×2 profiles = 2880 simulations)

Appendix C

List of Publications

1. P. Bagga, N. Paoletti, B. Alrayes, K. Stathis. (2020) '*Deep Reinforcement Learning Approach to Concurrent Bilateral Negotiation*'. Published in the proccedings of the *29th International Joint Conference on Artificial Intelligence (IJCAI 2020), Yokohama, Japan.*

2. P. Bagga, N. Paoletti, K. Stathis. (2021) '*Pareto Bid Estimation for Multi-Issue Bilateral Negotiation under User Preference Uncertainty*'. Published in the proceedings of the *30th IEEE International Conference on Fuzzy Systems (Fuzz-IEEE 2021)*, Luxembourg.

3. P. Bagga, N. Paoletti, B. Alrayes, K. Stathis. (2021) '*ANEGMA: an Automated NEGotiation model for e-MArkets*'. Published in the *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS).*

4. P. Bagga, N. Paoletti, K. Stathis. (2022) '*Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation*'. Accepted in the proceedings of the *21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022).*