Supporting the Design of General Automated Negotiators*

Raz Lin[1] and Sarit Kraus[1,2]
[1] Department of Computer Science
Bar-Ilan University
Ramat-Gan, Israel 52900
{linraz,sarit}@cs.biu.ac.il

[2] Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742 USA

Dmytro Tykhonov, Koen Hindriks and
Catholijn M. Jonker
Man-Machine Interaction Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
{d.tykhonov,k.v.hindriks,C.M.Jonker}@tudelft.nl

ABSTRACT

The design of automated negotiators has been the focus of abundant research in recent years. However, due to difficulties involved in creating generalized agents that can negotiate in several domains and against human counterparts, many automated negotiators are domain specific and their behavior cannot be generalized for other domains. Some of these difficulties arise from the differences inherent within the domains, the need to understand and learn negotiators' diverse preferences concerning issues of the domain and the different strategies negotiators can undertake. In this paper we present a system that enables alleviation of the difficulties in the design process of general automated negotiators termed GENIUS, a **G**eneral **E**nvironment for **N**egotiation with **I**ntelligent multi-purpose **U**sage **S**imulation. With the constant introduction of new domains, e-commerce and other applications, which require automated negotiations, generic automated negotiators encompass many benefits and advantages over agents that are designed for a specific domain. Based on experiments conducted with automated agents designed by human subjects using GENIUS we provide both quantitative and qualitative results to illustrate its efficacy. Our results show the advantages and underlying benefits of using GENIUS for designing general automated negotiators.

1. INTRODUCTION

One cannot understate the importance of negotiation and the centrality it has taken in our everyday lives, in general, and in specific situations in particular (e.g., hostage crises [20]). The fact that negotiation covers many aspects of our lives has led to extensive research in the area of automated negotiators, that is, automated agents capable of negotiating with other agents in a specific environment. However, when reviewing many of the agents suggested in the literature (e.g., [4, 5, 18]), one cannot ignore the fact that most of them lack two key fundamental features, which are, to our belief, most important for the design of successful general automated negotiators.

The first problem emerges from the inherent design of the automated negotiator. While humans can negotiate in different settings and domains, when designing an automated agent a decision should be made whether the agent should be a general purpose negotiator, that is, able to successfully negotiate in many settings and domain-independent (e.g., Lin *et al.* [23]), or suitable for only one specific domain (e.g., Ficici and Pfeffer [6] for the Colored Trail domain, or Kraus and Lehmann [19] for the Diplomacy game). There are obvious advantages for an agent's specificity in a given domain. It allows the agent's designer to construct better strategies that could allow it to negotiate better, in comparison to a more general purpose negotiator. However, this is also one of the major weaknesses of these type of agents. With the constant introduction of new domains, e-commerce and other applications, which require negotiations, the generality of an automated negotiator becomes important, as automated agents tailored to specific domain are useless since they cannot be used in the new domains and applications.

The second problem is that automated negotiators should work in open environments. Open environments lack a central mechanism for controlling the agents' behavior, and they may encounter human decision-makers whose behavior is diverse, cannot be captured by a monolithic model, make mistakes, is affected by cognitive, social and cultural factors, etc. [1, 21]. Examples of such environments include online markets, patient care-delivery systems, virtual reality and simulation systems used for training (e.g., the Trading Agent Competition (TAC) [32]).

While the two aforementioned difficulties (and proposed solutions) should be dealt with in more detail, in this paper we do not focus on the design of an efficient automated negotiator; we do not even claim that we have the right "formula" to do so. We do, however, present a tool that aims to help facilitate the *design* and *evaluation* of automated negotiators' strategies. The tool, GENIUS, is a **G**eneral **E**nvironment for **N**egotiation with **I**ntelligent multi-purpose **U**sage **S**imulation. To our knowledge, this is the first tool of its kind that both assists in the *design* of strategies for automated negotiators and also *supports* the evaluation process of the agent. Thus, we believe this tool is very useful for agent designers and can take a central part in the process of designing automated agents. While designing agents can be done in any agent oriented software engineering methodology, GENIUS wraps this in an easy-to-use environment and allows the designers to focus on the development of *strategies* for negotiation in an open environment with multi-attribute utility functions.

GENIUS incorporates several mechanisms that aim to support the design of a general automated negotiator. The first mechanism is

*This research is based upon work supported in part under NSF grant 0705587 and by the U. S. Army Research Laboratory and the U. S. Army Research Office under grant number W911NF-08-1-0144.

an analytical toolbox, which provides a variety of tools to analyze the performance of agents, the outcome of the negotiation and its dynamics. The second mechanism is a repository of domains and utility functions. Lastly, it also comprises repositories of automated negotiators. A comprehensive description of the tool is provided in Section 3.

In addition, GENIUS enables the evaluation of different strategies used by automated agents that were designed using the tool. This is an important contribution as it allows researchers to empirically and *objectively* compare their agents with others in different domains and settings. This is an important contribution with respect to the validation of results reported by researchers with regard to their automated negotiators.

In order to verify its efficacy, GENIUS was introduced to students, who were required to design automated agents for different negotiation tasks. Their agents were evaluated and both quantitative and qualitative results were gathered. A total of 65 automated agents were designed by 65 students. We describe the experimental methodology and results in Section 4. The results support our claim that GENIUS helps and supports the design process of an automated negotiator, from the initial design, through the evaluation of the agent, and re-design and improvements, based on its performance.

We begin by reviewing related research with respect to the design of general automated negotiators.

2. RELATED WORK

Research on general agent negotiators has given rise to a broad variety of such agents. The strategies of the agents usually vary from equilibrium strategies, optimal approaches and heuristics. Here we focus in particular on agents that are able to conduct bilateral negotiations with incomplete information. Examples of such general agent negotiators in the literature include, among others, Sycara *et al.* [30], who introduce a generic agent called *Bazaar*, Faratin *et al.* [4], who propose an agent that is able to make trade-offs in negotiations and motivated by maximizing the joint utility of the outcome (that is, the agents are utility maximizers that seek Pareto-optimal agreements), Karp *et al.* [15], who take a game-theoretic view and propose a negotiation strategy based on game-trees, Jonker *et al.* [14], who propose a negotiation model called *ABMP*, and Lin *et al.* [23], who propose an agent negotiator called *QOAgent*. All of these agents are proposed as agent negotiators that perform well in different domains, i.e. are domain-independent; for an example of an agent negotiator targeted at a particular negotiation domain, see Li *et al.* [22]. The motivation for introducing these agents, however, has varied and has related to diverse topics , such as learning in negotiation, the use of various heuristics, or negotiating with humans. Typically, alternating offer protocols are used where agents exchange offers in turn [29], sometimes with minor modifications as for example Lin *et al.* [23] proposed. Lomuscio *et al.* [24] in their work, offer useful classification of types of agent negotiators. Nonetheless, the important issue of the evaluation of agents' strategies and comparing between different strategies even in the same environment has not been adequately addressed by these researchers.

As we argue that it is useful to have a generic environment for designing and evaluating agent negotiators, we briefly review related work that is explicitly aimed at the evaluation of various agent negotiators. Most of the work reported herein concerns the evaluation of various *strategies* for negotiation used by such agents. Although some results were obtained by game-theoretic analysis (e.g. [17, 28]), most results were obtained by means of *simulation* (e.g. [2, 5, 8]). Devaux *et al.* [2] present work comparing agents negotiating in internet agent-based markets. In particular, they compare a strategy of their own agent with behavioral based strategies taken from the literature [3]. The simulations are performed with an abstract domain where agents need to negotiate the price of a product. Similarly, Henderson *et al.* [8] present results of a comparison of various negotiation strategies' performance in a simulated car hire scenario. Finally, Matos *et al.* [26] conducted experiments to determine the most successful strategies using an evolutionary approach in an abstract domain called the *service-oriented domain*.

Even though several of the approaches mentioned use a rather abstract domain with a range of parameters that may be varied, we argue that the focus on a single domain in most simulations is restrictive. A similar argument to this end has been put forward in [12]. The analysis of agent negotiators in multiple domains may significantly improve the performance of such agents. To the best of our knowledge, this is the first time that quantitative and qualitative evidence is presented to substantiate this claim.

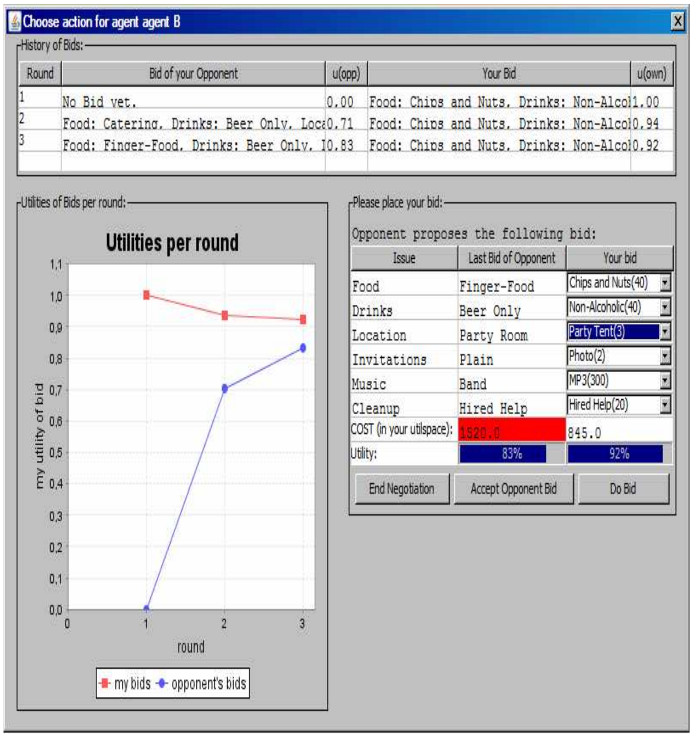
Manisterski *et al.* [25] discuss how people who design agent negotiators change their design over time. They study how students changed their design of a trading agent that negotiates in an open environment. After initial design of their agents, human designers obtained additional information about the performance of their agents by receiving logs of negotiations between their agents and agents designed by others. These logs provided the means to analyze the negotiation behavior, and an opportunity to improve the performance of the agents. The GENIUS environment discussed here provides a tool that supports such analysis, subsequent improvement of the design, and structures the enhancement process.

With regard to systems that facilitate the actual design of agents or agent strategies in negotiations, few systems are close to our line of suggested work. Most of the systems that can be somewhat related to the main focus of our paper are negotiation support systems (e.g., the Interactive Computer-Assisted Negotiation Support system (ICANS) [31], the InterNeg Support Program for Intercultural REsearch (INSPIRE)), however, they do not deal with the combination of both the evaluation of strategies and the facilitation of automated negotiator's design. INSPIRE [16] is a Web-based negotiation support system, which primary goal is to facilitate negotiation research in an international environment. The system enables negotiation between two humans and collects data about negotiations and has some basic functionality for the analysis of the agreements, such as calculation of the utility of an agreement and exchanged offers. However, it does not allow integration of an automated negotiating agent and thus does not include repositories of agents as we propose. Perhaps Neg-o-Net [7] is the most similar to GENIUS than all other support systems. The Neg-o-Net model is a generic agent-based computational simulation model for capturing multi-agency negotiations concerning resource and environmental management decisions. Neg-o-Net model includes both negotiation algorithm and agent models. Agent's preferences are modeled using digraphs (scripts). Nodes represent states of the agent that can be achieved by performing actions (arcs). Each state is evaluated using utility functions. The user can modify agent's script to model his/her preferences w.r.t. states and actions. Yet, their system does not allow for the incorporation of human negotiators, but only automated ones. Moreover, they do not provide any evaluation mechanism of the strategies as GENIUS provides.

We continue with a detailed description of the GENIUS system, followed by the experiments we conducted and the results.

3. THE GENIUS SYSTEM

GENIUS is a **G**eneral **E**nvironment for **N**egotiation with **I**ntelligent multi-purpose **U**sage **S**imulation. The aim of the tool is to facili-

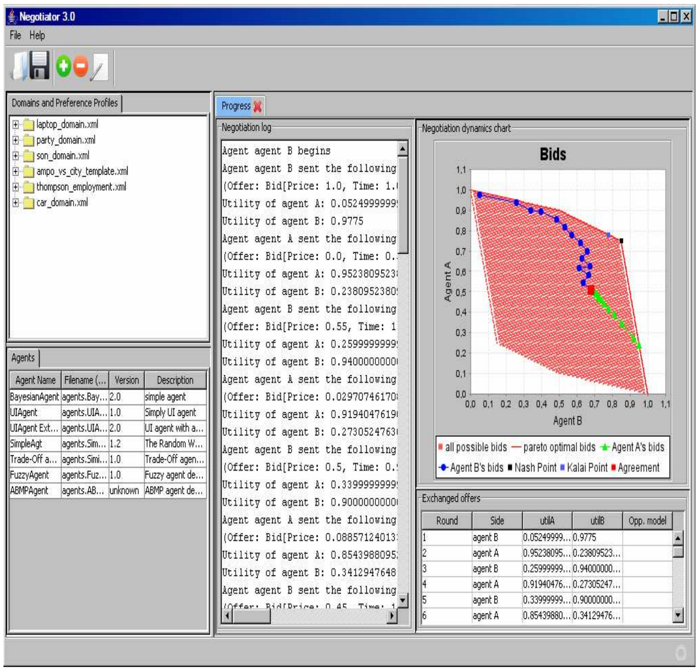


Round	Bid of your Opponent	u(opp)	Your Bid	u(own)
1	No Bid yet.	0.00	Food: Chips and Nuts, Drinks: Non-Alco	1.00
2	Food: Catering, Drinks: Beer Only, Loca	0.71	Food: Chips and Nuts, Drinks: Non-Alco	0.94
3	Food: Finger-Food, Drinks: Beer Only, I	0.83	Food: Chips and Nuts, Drinks: Non-Alco	0.92

Opponent proposes the following bid:

Issue	Last Bid of Opponent	Your bid
Food	Finger-Food	Chips and Nuts(40)
Drinks	Beer Only	Non-Alcoholic(40)
Location	Party Room	Party Tent(3)
Invitations	Plain	Photo(2)
Music	Band	MP3(300)
Cleanup	Hired Help	Hired Help(20)
COST (in your utilspace):	1520.0	845.0
Utility:	83%	92%

Figure 1: An example of the GUI interface of GENIUS for human negotiators during a specific negotiation session.



- laptop_domain.xml
- party_domain.xml
- son_domain.xml
- ampo_vs_city_template.xml
- thompson_employment.xml
- car_domain.xml

Agent Name	Filename (...	Version	Description
BayesianAgent	agents.Bay...	2.0	simple agent
UIAgent	agents.UIA...	1.0	Simply UI agent
UIAgent Ext...	agents.UIA...	2.0	UI agent with a...
SimpleAgt	agents.Sim...	1.2	The Random W...
Trade-Off a...	agents.Simi...	1.0	Trade-Off agen...
FuzzyAgent	agents.Fuz...	1.0	Fuzzy agent de...
ABMPAgent	agents.AB...	unknown	ABMP agent de...

```
Agent agent B begins
Agent agent B sent the following
(Offer: Bid[Price: 1.0, Time: 1.
Utility of agent A: 0.0524999999
Utility of agent B: 0.9775
Agent agent A sent the following
(Offer: Bid[Price: 0.0, Time: 0.
Utility of agent A: 0.9523809523
Utility of agent B: 0.2380952380
Agent agent B sent the following
(Offer: Bid[Price: 0.55, Time: 1
Utility of agent A: 0.2599999999
Utility of agent B: 0.9400000000
Agent agent A sent the following
(Offer: Bid[Price: 0.0297074617
Utility of agent A: 0.9194047619
Utility of agent B: 0.2730524763
Agent agent B sent the following
(Offer: Bid[Price: 0.5, Time: 0.
Utility of agent A: 0.3399999999
Utility of agent B: 0.9000000000
Agent agent A sent the following
(Offer: Bid[Price: 0.0885712401
Utility of agent A: 0.8543988095
Utility of agent B: 0.3412947648
Agent agent B sent the following
(Offer: Bid[Price: 0.45, Time: 1
```

Round	Side	utilA	utilB	Opp. model
1	agent B	0.05249999...	0.9775	
2	agent A	0.95238095...	0.23809523...	
3	agent B	0.25999999...	0.94000000...	
4	agent A	0.91940476...	0.27305247...	
5	agent B	0.33999999...	0.90000000...	
6	agent A	0.85439880...	0.34129476...	

Figure 2: An example of GENIUS' main user interface, showing the results of a specific negotiation session.

tate the design of negotiation strategies. Using GENIUS programmers can focus mainly on the strategy design. This is achieved by GENIUS by providing both a flexible and easy-to-use environment for implementing agents and mechanisms that support the strategy design and analysis of the agents.

GENIUS enables negotiation between automated agents, as well as humans. Human negotiators and automated ones can be joined in a single negotiation session. Human negotiators interact with GENIUS via a graphical user interface (GUI). GUIs included in GENIUS allow the human negotiator to exchange offers with his/her counterpart, to keep track of them, and consult with his/her own preference profile (that is, a utility score assigned to each issue of the negotiation) to evaluate the offers. Figure 1 shows an example of a human negotiator GUI. For automated agents, GENIUS provides skeleton classes to help designers implement their negotiating agents. It provides functionality to access information about the negotiation domain and the preference profile of the agent. An interaction component of GENIUS manages the rules of encounter or protocol that regulates the agent's interaction in the negotiation. This allows the agent designer to focus on the design of the agent, and eliminates the need to implement the communication protocol or the negotiation protocol. Existing agents can be easily integrated in the GENIUS by means of adapters[1].

GENIUS provides a flexible simulation environment. A researcher can setup a single negotiation session or a tournament via the GUI simulation (see Figure 2) using the negotiation domains and preference profiles from a repository (top left corner of the GUI simulation), and choose strategies for the negotiating parties (top bottom corner of the GUI simulation). For this purpose, a graphical user interface layer provides options to create a negotiation domain, defines agent preferences, allows human user(s) to participate in a negotiation, and reviews performance and benchmark results of agents that conducted a negotiation. This also includes defining different preferences for each role.

A negotiation domain is a specification of the objectives and issues to be resolved by means of negotiation. Objectives allow to define a tree-like structure with either other objectives again or issues as children. Various types of issues are allowed, including discrete enumerated value sets, integer-valued sets, real-valued sets, as well as a special type of issue called price issue. Additionally, a specification of a negotiation domain may introduce constraints on acceptable outcomes.

A preference profile specifies the preferences regarding possible outcomes of an agent. This can be considered a function mapping outcomes of a negotiation domain on the level of satisfaction of an agent associated with that outcome. The structure of a preference profile for obvious reasons resembles that of a domain specification. The tree-like structure enables specification of relative priorities of parts of the tree.

Seven negotiation domains are currently collected in the repository of GENIUS. Each domain has at least two preference profiles required for bilateral negotiations. The number of issues in the domains ranges from 3 to 10, where the largest negotiation domain in the repository is the AMPO vs City taken from [27], and has over 7,000,000 possible agreements. Issues in the repository have different predictabilities of the evaluation of alternatives. Issues are considered predictable when even though the actual evaluation function for the issue is unknown, it is possible to guess some of its global properties (for more details, see [12]). The repository of strategies currently contains six automated negotiation strategies, such as the ABMP strategy [13], the Zero-Intelligence strategy [9], the QO-strategy [23], the Bayesian strategy [11] and others. The

[1] Indeed as was shown in [10].

repositories of domains and of agents allow agent designers test their agents on the different domains and against different kind of agents and strategies.

GENIUS provides an analytical toolbox for evaluating negotiation strategies. The toolbox calculates optimal solutions, such as the Pareto efficient frontier, Nash product and Kalai-Smorodinsky [27]. These solutions are visually shown to the negotiator or the designer of the automated agent, as depicted in the top right corner of Figure 2. We can see all the possible agreements in the domain (all dotted areas) where the highest and most right lines denote the Pareto efficient frontier. During the negotiation each side can see the distance of its offers from this Pareto frontier as well as the distance from previous offers (as shown by the two lines inside the curve).

Using the analytical toolbox one can analyze the dynamic properties of a negotiation session, such as a classification of negotiation moves (a step-wise analysis of moves) and the sensitivity to a counterpart's preferences measure, as suggested by Hindriks *et al.* [9]. For example, one can see whether his/her strategy is concession oriented, i.e., steps are intended to be concessions, but in fact some of these steps might be *unfortunate*, namely, although from the receiver's perception the proposer of the offer is conceding, the offer is actually worse than the previous offer. The result of the analysis can help agent designers improve their agents.

4. EXPERIMENTS

The experiments described below were conducted in order to test the efficacy of the mechanisms incorporated into GENIUS. Prior to these experiments we verified that GENIUS indeed facilitates the flexible creation of tournaments. As an example, in [9] we evaluated several negotiation strategies in a tournament setup where every negotiation strategy had to negotiate on several different negotiation domains with various preference profiles and against a range of negotiation strategies used by different opponents. As a result, we found that negotiation strategies that are designated as generic and are meant to perform well independent of the domain, nevertheless may be inefficient in particular negotiation setups. For example, the Trade-Off strategy, introduced in [4], shows excellent performance when confronted with itself but its performance is not as good when negotiating against an agent that uses a suboptimal strategy. Furthermore, evidently the characteristics of the negotiation domain and preference profiles, such as the number of issues, the opposition of the preferences and their predictability [9, 12], play a significant role in the performance of negotiation strategies. These results were obtained with the help of the analytical toolbox in GENIUS using GENIUS's repositories of domains, preference profiles, and strategies.

In the experiments we present in this paper, human subjects were instructed to design automated agents that will negotiate with other automated agents in a tournament in an open environment. The experiments were conducted in several phases in order to validate the results. These experiment results show that GENIUS indeed supports the design of general automated negotiators. In the following subsections we describe the negotiation domains, the experimental methodology and we review the results. We begin by presenting the negotiation domains.

4.1 Experimental Domain

While the first experiment was only run on one domain, the second experiment was run on three domains. In the first two domains we modeled three possible agent types, and thus a set of six different utility functions was created for each domain. In the third domain only one type was possible for the different roles. The different types of agents describe the different approaches towards the negotiation process and the other party. For example, the different approaches can describe the importance each agent associates with the effects of the agreement over time. One agent might have a long term orientation regarding the final agreement. This type of agent would favor agreements concerned more with future outcomes of the negotiations, than those focusing only on solving the present problem. On the other hand, another agent might have a short term orientation which focuses on solving only the burning issues under negotiation without dealing with future aspects that might arise from the negotiation or its solutions. Finally, there can also be agents with a compromise orientation. These agents try to find the middle grounds between the possible agreements.

Each negotiator was assigned a utility function at the beginning of the negotiations but had incomplete information regarding the counterpart's utility. That is, the different possible types of the counterpart were public knowledge, but the exact type of the counterpart was unknown.

We describe the three domains in the following subsections. The first two domains are taken from [23], in which they were used for negotiations by human negotiators as well as automated ones. The third domain is taken from the Dispute Resolution Research Center at Kellogg School of Management.

4.1.1 The World Health Organization's Framework Convention on Tobacco Control Domain

In this scenario England and Zimbabwe negotiate in order to reach an agreement evolving from the World Health Organization's Framework Convention on Tobacco Control, the world's first public health treaty. The principal goal of the convention is "to protect present and future generations from the devastating health, social, environmental and economic consequences of tobacco consumption and exposure to tobacco smoke."

The leaders of both countries are about to meet at a long scheduled summit. They must reach an agreement on 4 issues, each with several attributes:

1. The total amount to be deposited into the Global, Tobacco Fund to aid countries seeking to rid themselves of economic dependence on tobacco production;

2. Impact on other aid programs;

3. Trade issues;

4. Creation of a forum to explore comparable arrangements for other long-term health issues.

Consequently, a total of 576 possible agreements exist in this domain. While for the first two issues there are contradictory preferences for England and Zimbabwe, for the last two issues there are options which might be jointly preferred by both sides.

4.1.2 The Job Candidate Domain

In this scenario, a negotiation takes place after a successful job interview between an employer and a job candidate. In the negotiation both the employer and the job candidate wish to formalize the hiring terms and conditions of the applicant. In contrast to the England-Zimbabwe scenario, some issues *must* be agreed upon to achieve even a partial agreement. Below are the issues under negotiation:

1. Salary;

2. Job description;

3. Social benefits;

4. Promotion possibilities;

5. Working hours.

In this scenario, a total of 1,296 possible agreements exist.

4.1.3 The Class Project Domain

In this scenario, Bob and Alice need to decide on a final project plan. In contrast to the other two domains, in this domain the utility preferences of both sides are completely symmetric. For each issue, five possible values are negotiable. The issues under negotiation are:

1. Project's topic;

2. Project's type;

3. Method of presentation;

4. Completion time;

5. Preparation time;

6. Meeting times.

This is also the largest scenario of all three, in terms of possible agreements. In this scenario, a total of 15,625 possible agreements exist. Yet, unlike the previous domains, only one type for each role was possible.

4.2 Experimental Methodology

We evaluated the process of the agents design by requiring computer science undergraduate and graduate students to design automated agents. These agents were matched twice in a tournament with all other agents. After each tournament, the students were exposed to one of the mechanisms of GENIUS and were allowed to re-design their agent. Then, they were matched again in a tournament. In addition, after the students submitted their new agents, they were required to fill in questionnaires and evaluate the design process of their agents.

We conducted two experiments. In the first, we evaluated the efficacy of the analytical toolbox. The second experiment was designed to enable evaluation of the efficacy of the repositories of domains and repositories of agents. We describe both experiments in the following subsections.

4.2.1 Evaluation of the Analytical Toolbox

In the first experiment, 51 undergraduate students were required to design an automated negotiator using the GENIUS environment. The students were instructed to design an automated negotiator which will be able to negotiate in several domains, however, they were only given the Job Candidate domain described in Section 4.1.2 as an example. In addition, three automated negotiators were supplied with the tool[2]:

1. An agent that follows the Bayesian strategy [11];

2. Another automated agent that follows the Agent-Based Market Places (ABMP) strategy, which is a concession-oriented negotiation strategy [13], though, the strategy itself was not explained to the students;

3. A simple agent that sorts all possible offers according to their utility and sends them one-by-one to the opponent starting with the highest utility.

In the first phase, the students were unaware of the analytical toolbox (which was also removed from the environment and the code). After the students submitted their agent, they were given an upgraded environment which included the analytical toolbox. They were given an explanation about its features. Then they were allocated several days in which they could use it to re-design their agent.

The students' agents were evaluated three times. The first time included running the first phase agents against all other agents. Thus, each agent was matched against all 51 agents (including itself), each time under a different role. That is, each agent participated in 102 negotiations, and a total of 5,202 simulations were executed. The second time, each revised agent was matched against all 51 revised agents (including itself). This allowed us to validate the efficacy of the analytical toolbox by comparing the performance of each revised agent to its original performance. The third time included running the revised agents against each other using a new domain, the England-Zimbabwe domain, which they were unaware of during the design process. This allowed us to evaluate whether the analytical toolbox by itself is or is not suffice for designing generalized agents.

4.2.2 Evaluation of the Domain and Agent Repositories

In this experiment, like the previous experiment, 14 graduate students were required to design an automated negotiator using the GENIUS environment. They were also instructed that their task is to design an efficient negotiator that will be matched with all other automated negotiators. Throughout the design process they were unaware of the analytical toolbox. In the first part of the exercise they were given the Job Candidate domain as an example. After their submissions, they were given an additional domain, the England-Zimbabwe domain described in Section 4.1.1. As in the previous experiment, they were allocated several days in which they could re-design their agents based on the new introduced domain. Furthermore, half of the students were given logs of all their matches during the tournament. The logs included detailed information of the negotiation process.

In this experiment the students' agents were evaluated four times. The first time included running the first phase agents against all other agents. Thus, each agent was matched against all 14 agents (including itself). The agents were run twice. Once on the domain that was known to them during the design of the original agents, i.e., the Job Candidate domain, and once in the England-Zimbabwe domain which they were unaware of at the time. The second time, each revised agent was matched against all 14 revised agents in the Job Candidate domain and in the England-Zimbabwe domain, respectively. This allowed us to validate the efficacy of both the introduction of a new domain and the usage of logs of past negotiations by comparing the performance of each revised agent to its original performance. Lastly, we ran the students' agents against each other using a new domain, the Class Project domain, which the designers were unaware of during the entire design process. Again, we ran both the original agents and the revised agents. This allowed us to evaluate whether or not the two given domains were suffice for designing efficient generalized agents.

4.3 Experimental Results

The main goal of the experiments was to verify that the mechanisms in GENIUS assist in alleviating the difficulties in designing

[2] The agents were supplied with their code to also demonstrate to the students the use of skeleton classes.

efficient general automated negotiators.

As we mentioned earlier, we experimented in three distinct domains. The utility values ranged from -575 to 895 for the England role and from -680 to 830 for the Zimbabwe role; in the Job Candidate domain from 170 to 620 for the employer role and from 60 to 635 for the job candidate role, and in the Class Project domain from 0 to 29,200 for both sides.

4.3.1 Experiments with the Analytical Toolbox

We evaluate the design of the agents using both quantitative results and qualitative results. The quantitative results, presented in Table 1, comprise a comparison of the agents' performance in the different settings of the experiments, while the qualitative results are gathered from the questionnaires the subjects filled in after the submission of the revised agents.

Approach/Role	Employer	Job Candidate
Original Agents	517	490
Revised Agents	525	505

Table 1: Average utility values gained by the automated agents before and after being exposed to the analytical toolbox.

The average utility gained by all the revised agents was 525 when playing the role of the employer and 505 when playing the role of the job candidate. These averages are significantly higher (using t-test with p-value < 0.001) in both roles as compared to the average utilities of the original agents (517 and 490, respectively).

In order to assess the ease of use of the GENIUS environment in creating generalized agents, as well as how helpful the analytical toolbox was, the students were asked to answer several questions on a questionnaire they were administered. 67% of the students indicated that they re-designed their agent in the second part, after being introduced to the analytical toolbox, and 79.6% used it to gain a better understanding of the negotiation and to redesign their agents. Moreover, in a scale of 1 (being the lowest) to 7 (being the highest), the students rated the helpfulness of the tool in understanding the dynamics of the negotiation and the strategy of their agent with an average of 4.06. The students indicated that the tool enabled them to attain a clearer view of the negotiation dynamics by visualizing the spectrum of offers and their utilities, and understand which offers to accept and which offers to propose. Some students also commented that the tool helped them verify that their implemented strategy was indeed as they had intended it to be. Figure 3 presents the total rating the students gave for the helpfulness of the analytical toolbox.

It is interesting to note that most students indicated that they designed their agent to play as if they were the negotiator (an average score of 4.54), yet they also indicated that the fact that they knew that their counterpart would be a computer agent and not a human affected their strategy as they tried to take advantage of this fact.

While this encouraged us as to the efficacy of the analytical toolbox as a supporting mechanism for designing automated negotiators, we still had to verify whether it could also assist in the design of generalized automated negotiators. To test the generality of the agents, we ran the revised agents in a new domain, the England-Zimbabwe domain, of which the students were unaware. However, in this domain only 32.3% of the negotiations were completed successfully, i.e., with a full agreement, as compared to almost double the amount of negotiations that were completed successfully on the known domain (64.4%). That is, while the analytical toolbox was indeed helpful to the students and assisted them in the design of their agent, it was not suffice in order to help them design an efficient general agent. Thus, we continued to devise a second experiment with repositories of domains and agents. The results of this experiment are described in the next subsection.

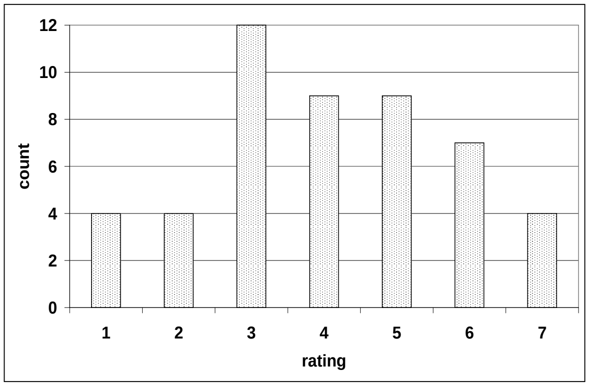


Figure 3: Rating of the helpfulness of the analytical toolbox.

4.3.2 Experiments with Repositories of Domains and Agents

We continued to test other aspects of GENIUS to see whether they help in the design process of agents' strategies. In this experiment, the domains also had a time effect. That is, costs were assigned to each agent, such that during the negotiation process, the agents might gain or lose utility over time. The results are summarized in Tables 2 and 3. In the first part, the students were required to design a general agent, however only one domain was given to them. The average utility scores of their agents in the Job Candidate domain were 363 for the Employer role and 336.8 for the Job Candidate role. In order to evaluate the improvement of the agents due to the logs of past negotiations in which they were matched with all other agents, we continued to run the students' revised agents in the same domain. The results of the agents in this experiment were better, yet not statistically significant (an average utility of 384.29 with a p-value < 0.07 and 365.78 with a p-value < 0.06 for the Employer and the Job Candidate roles, respectively). In addition, significantly more negotiations ended with a full agreement (77.3% in the first stage, as compared to 85% in the second stage, p-value < 0.05).

With respect to using the repositories of agents as a means of improving an agent's strategy, 80% of the students who received the logs of their agents' past negotiations indicated that they indeed used it to improve their agents' behavior. Some noticed, thanks to the logs, that they had bugs in their strategy or that their agents' behavior was too strict and less compromising, causing too many negotiations to end with opting-out. Using this insight, they revised their agents' behavior.

To evaluate the benefits of the repositories of domains on the performance of their agent, we first matched the students' original agents against each other in the new England-Zimbabwe domain. Recall that the original agents were designed without knowledge about the new domain. We then compared these results with the results of the revised agents that had knowledge of the new domain. The average utility scores of the original agents were 302.11 for the

Approach/Role	Employer	Job Candidate
Original Agents	363	336.8
Revised Agents	384.29	365.78

Table 2: Average utility values gained by the automated agents before and after being exposed to logs of past negotiations.

England-Zimbabwe Domain		
Approach/Role	England	Zimbabwe
Original Agents	302.11	-413.57
Revised Agents	369.99	-377.37
Class Project Domain		
Approach/Role	Bob	Alice
Original Agents	11,357	10,655
Revised Agents	13,348	12,113

Table 3: Average utility values gained by the automated agents before and after being exposed to an additional domain.

England role and -413.57 for the Zimbabwe role. The results of the revised agents were significantly better in the case of England (an average utility of 369.99 with a p-value < 0.03), while the utility was better, though not statistically significant, for the role of Zimbabwe (-377.37). However, with the revised agents significantly more negotiations ended with a full agreement (39.2% in the first stage, as compared to 50.5% in the second stage, p-value < 0.02).

To validate these results, the students' agents were then run in the Class Project domain, described in Section 4.1.3, of which they were unaware during their entire design process. We first ran the original agents in that domain, and the average utility scores of the agents were 11,357 for Bob's role and 10,655 for the Alice's role. In addition, only 66.5% of the negotiations ended with a full agreement. We then ran their revised agents against themselves. Consequently, significantly more negotiations ended with a full agreement (76.8%, p-value < 0.02), resulting also in higher average utility values of 13,348 for Bob and 12,113 for Alice. When the agents played the role of Bob these results were also significant (p-value < 0.04). We believe that if we had more students' designed agents the average utility values the agents achieved could have been significantly better in both roles, both in the Class Project domain and in the England-Zimbabwe domain.

In this set of experiments we also gave the students questionnaires to help qualitatively assess the efficiency of the repositories of domains and agents. The students had to rate several statements in a scale of 1 (being the lowest) to 7 (being the highest). The students indicated that their agent was more generic after the second domain was introduced. The average score for the agent's generality in the first stage was 5.38 compared to 6.08 for the revised version. Overall, the students rated their agents' generality as 6.0, and they asserted that their agents would succeed in playing well in other domains as well, with an average rating of 5.38.

5. CONCLUSIONS

Availability of efficient general automated negotiators has two main advantages. Firstly, it minimizes the effort required for adaptation of a general automated negotiator to a new domain. Furthermore, the general automated negotiator can be used as a starting point to create a more efficient negotiator that takes into account a domain specific knowledge, e.g., available a priori information about the most likely preferences of the opponent. Secondly, a general automated negotiated agent is not biased towards domain specific features that can have negative influence on its negotiation efficiency.

This paper presents a simulation environment which supports the design of general automated negotiators. Extensive simulations with more than 60 computer science students were conducted to validate the efficacy of the simulation environment. The results show that Genius indeed supports the design of general automated negotiators, and even enables the designers to improve their agents' performance while retaining their generality. This is an important feature, since most of the time general automated negotiators are perceived to perform worse than agents designed specifically for a given domain.

We conducted experiments with automated agents in three distinct domains. The largest domain comprised more than 15,000 possible agreements. While this proves that the simulation environment supports repositories of domains, we did not evaluate the agents on very large domains (e.g., more than 1,000,000 agreements). Many of the automated agents the students designed took advantage of the small domains and reviewed all possible agreements. This would be infeasible in larger domains with a deadline for the negotiation or each turn in the negotiation.

Another issue for future research is the use of Genius for the design of automated negotiators that can successfully and efficiently negotiate with human negotiators. As we mentioned, some of the students took advantage of the fact that they were aware that their agents would be matched only with other automated agents. It would be interesting to evaluate the performance of their agents against human negotiators as well.

In future work, we plan to run complete tournaments between the agents in the repository on all available negotiation domains. This would allow us to identify the most efficient strategy currently available in the repository. In addition, we believe that efficiency of a negotiation strategy can depend on the opponent's strategy as well as on the characteristics of the negotiation domain and preference profiles. The analytical toolbox of Genius would allow us to identify such dependencies and understand the reasoning behind them. Logs of negotiation sessions produced by Genius can be used to discover patterns of negotiation behavior of the automated negotiation strategies of human negotiators.

6. REFERENCES

[1] M. H. Bazerman and M. A. Neale. Negotiator rationality and negotiator cognition: The interactive roles of prescriptive and descriptive research. In H. P. Young, editor, *Negotiation Analysis*, pages 109–130. The University of Michigan Press, 1992.

[2] L. Devaux and C. Paraschiv. Bargaining on an internet agent-based market: Behavioral vs. optimizing agents. *Electronic Commerce Research*, 1:371–401, 2001.

[3] P. Faratin, C. Sierra, and N. R. Jennings. Negotiation decision functions for autonomous agents. *Int. Journal of Robotics and Autonomous Systems*, 24(3-4):159–182, 1998.

[4] P. Faratin, C. Sierra, and N. R. Jennings. Using similarity criteria to make trade-offs in automated negotiations. *AIJ*, 142(2):205–237, 2002.

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

[6] S. Ficici and A. Pfeffer. Modeling how humans reason about others with partial information. In *Proceedings of AAMAS'08*, pages 315–322, 2008.

[7] D. Hales. Neg-o-net - a negotiation simulation test-bed. Technical Report CPM-02-109, CPM, April 2002.

[8] P. Henderson, S. Crouch, R. J. Walters, and Q. Ni. Comparison of some negotiation algorithms using a tournament-based approach. In *Agent Technologies, Infrastructures, Tools, and Applications for E-Services*, pages 137–150, 2008.

[9] K. Hindriks, C. Jonker, and D. Tykhonov. Negotiation dynamics: Analysis, concession tactics, and outcomes. In *Proceedings of IAT'07*, pages 427–433, 2007.

[10] K. Hindriks, C. Jonker, and D. Tykhonov. Towards an open negotiation architecture for heterogeneous agents. In *12th International Workshop on Cooperative Information Agents (CIA'08)*, volume 5180 of *LNAI*, pages 264–279. Springer, 2008.

[11] K. Hindriks and D. Tykhonov. Opponent modelling in automated multi-issue negotiation using bayesian learning. In *Proceedings of AAMAS'08*, pages 331–338, 2008.

[12] K. Hindriks and D. Tykhonov. Towards a quality assessment method for learning preference profiles in negotiation. In *Proceeding of Workshop on Agent-based Complex Automated Negotiations*, pages 64–71, 2008.

[13] C. Jonker and J. Treur. An agent architecture for multi-attribute negotiation. In *Proceedings of IJCAI'01*, pages 1195–1201, 2001.

[14] C. M. Jonker, V. Robu, and J. Treur. An agent architecture for multi-attribute negotiation using incomplete preference information. *JAAMAS*, 15(2):221–252, 2007.

[15] A. H. Karp, R. Wu, K. yut Chen, and A. Zhang. A game tree strategy for automated negotiation. Technical Report HPL-2003-154, 2003.

[16] G. E. Kersten and S. J. Noronha. Www-based negotiation support: design, implementation, and use. *Decision Support Systems*, 25(2):135–154, 1999.

[17] S. Kraus. *Strategic Negotiation in Multi-Agent Environments*. Cambridge: The MIT Press, 2001.

[18] S. Kraus, P. Hoz-Weiss, J. Wilkenfeld, D. R. Andersen, and A. Pate. Resolving crises through automated bilateral negotiations. *AIJ*, 172(1):1–18, 2008.

[19] S. Kraus and D. Lehmann. Designing and building a negotiating automated agent. *Computational Intelligence*, 11(1):132–171, 1995.

[20] S. Kraus, J. Wilkenfeld, M. A. Harris, and E. Blake. The hostage crisis simulation. *Simulation & Gaming*, 23(4):398–416, 1992.

[21] D. A. Lax and J. K. Sebenius. Thinking coalitionally: party arithmetic, process opportunism, and strategic sequencing. In H. P. Young, editor, *Negotiation Analysis*, pages 153–193. The University of Michigan Press, 1992.

[22] C. Li, J. Giampapa, and K. Sycara. A review of research literature on bilateral negotiations. Technical Report CMU-RI-TR-03-41, 2003.

[23] R. Lin, S. Kraus, J. Wilkenfeld, and J. Barry. Negotiating with bounded rational agents in environments with incomplete information using an automated agent. *AIJ*, 172(6-7):823–851, 2008.

[24] A. Lomuscio, M. Wooldridge, and N. R. Jennings. A classification scheme for negotiation in electronic commerce. In *Proceedings of AMEC'01*, pages 19 – 33, 2001.

[25] E. Manistersky, R. Lin, and S. Kraus. Understanding how people design trading agents over time. In *Proceedings of AAMAS'08*, pages 1593–1596, 2008.

[26] N. Matos, C. Sierra, and N. R. Jennings. Determining successful negotiation strategies: An evolutionary approach. In *Proceedings of ICMAS'98*, 1998.

[27] H. Raiffa. *The Art and Science of Negotiation*. Harvard University Press, 1982.

[28] J. Rosenschein and G. Zlotkin. *Designing Conventions for Automated Negotiation among Computers*. Cambridge: The MIT Press, 1994.

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

[30] K. Sycara and D. Zeng. Bayesian learning in negotiation. In *Working Notes of the AAAI Spring Symposium on Adaptation, Co-evolution and Learning in Multiagent Systems*, 1996.

[31] E. M. Thiessen, D. P. Loucks, and J. R. Stedinger. Computer-assisted negotiations of water resources conflicts. *GDN*, 7(2):109–129, 1998.

[32] M. P. Wellman, A. Greenwald, and P. Stone. *Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition*. MIT Press, Cambridge MA, USA, 2007.