Computational Intelligence

An International Journal

Computational Intelligence, Volume 0, Number 0, 2012

GENIUS: AN INTEGRATED ENVIRONMENT FOR SUPPORTING THE DESIGN OF GENERIC AUTOMATED NEGOTIATORS

Raz Lin,[1] Sarit Kraus,[1,2] Tim Baarslag,[3] Dmytro Tykhonov,[3] Koen Hindriks,[3] and Catholijn M. Jonker[3]

[1]*Department of Computer Science, Bar-Ilan University, Ramat-Gan, Israel*
[2]*Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland, USA*
[3]*Man-Machine Interaction Group, Delft University of Technology, Mekelweg, Delft, the Netherlands*

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. Finally, we also analyze a recent automated bilateral negotiators competition that was based on Genius. Our results show the advantages and underlying benefits of using Genius and how it can facilitate the design of general automated negotiators.

Received 28 November 2010; Revised 25 October 2011; Accepted 6 November 2011

Key words: agents competition, automated negotiation, human/computer interaction, bilateral negotiation.

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., in hostage crises situations as described in Kraus et al. (1992)). 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 setting.

There are several difficulties that emerge when designing automated negotiating agents, that is automated programs with negotiating capabilities. First, although people 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, domain-independent (e.g., Lin et al. 2008) and able to successfully negotiate in many settings or suitable for only one specific domain (e.g., Ficici and Pfeffer (2008) for the Colored Trail domain, or Kraus and Lehmann (1995) for the Diplomacy game). There are obvious advantages of an agent's specificity in a given domain. It allows the agent designers to construct strategies that enable better negotiation compared to strategies for a more general purpose negotiator. However, this is also one of the major weaknesses of these types of agents. With the constant introduction of new domains, e-commerce and other applications that require negotiations, the generality of an automated negotiator becomes important, because automated agents tailored to specific domains are useless when they are used in the new domains and applications.

Address correspondence to Raz Lin, Department of Computer Science, Bar-Ilan University, Ramat-Gan, Israel 52900; e-mail: razlin81@gmail.com

Another difficulty in designing automated negotiators concerns open environments, such as online markets, patient care delivery systems, virtual reality, and simulation systems used for training (e.g., the Trading Agent Competition (TAC) reported by Wellman, Greenwald, and Stone (2007)). These environments lack a central mechanism for controlling the agents' behavior, where agents may encounter human decision makers whose behavior is diverse. Such diverse behavior cannot be captured by a monolithic model; humans tend to make mistakes because they are affected by cognitive, social, and cultural factors, etc. (Bazerman and Neale 1992; Lax and Sebenius 1992).

Although the two aforementioned difficulties 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 an environment to facilitate the *design* and *evaluation* of automated negotiators' strategies. The environment, 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 environment 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 environment is very useful for agent designers and can take a central part in the process of designing automated agents. Although 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 multiattribute utility functions.

GENIUS incorporates several mechanisms that aim to support the design of a general automated negotiator. The first mechanism is 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. Finally, 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 and validate their results. This in turn allows to generate better automated negotiators, explore different learning and adaptation strategies and opponent models, and collect state-of-the-art negotiating agents, negotiation domains, and preference profiles. It also enables making them available and accessible for the negotiation research community.

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 redesign and improvements, based on its performance.

In May 2011 we organized the first automated negotiating agents competition (ANAC), with the aim of coordinating the research into automated agent design and proficient bidding strategies for bilateral multiissue closed negotiation, similar to the objective achieved by the TAC for the trading agent problem (Wellman et al. 2007). The entire competition was based on the GENIUS environment. We analyze the benefits of using GENIUS for this competition in Section 5.

We begin by reviewing related research relating 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 through optimal approaches to 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 and Zeng (1996), who introduce a generic agent called *Bazaar*, Faratin, Sierra, and Jennings (2002), who propose an agent that is able to make trade-offs in negotiations and is motivated by maximizing the joint utility of the outcome (that is, the agents are utility maximizers that seek Pareto-optimal agreements), Karp et al. (2003), who take a game-theoretic view and propose a negotiation strategy based on game-trees, Jonker, Robu, and Treur (2007), who propose a negotiation model called *Agent-Based Market Places* (*ABMP*), and Lin et al. (2008), who propose an agent negotiator called *QOAgent*. All of these agents are proposed as domain-independent agents. 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 people. Typically, alternating offer protocols are used where agents exchange offers in turn (Rubinstein 1982), sometimes with minor modifications, for example, Lin et al. (2008) proposed. Lomuscio, Wooldridge, and Jennings (2001) in their work, offer a useful classification of types of agent negotiators. Nonetheless, the importance and contribution of GENIUS is that it provides, in addition to the design of domain-independent agents, a general infrastructure for designing such agents, defining the domains and evaluating based on other agents developed in the same infrastructure. GENIUS was built in the intent to be publicly available in the aim of providing researchers a simple and effective tool for designing negotiations' strategies.

As we argue that a generic environment for designing and evaluating agent negotiators is useful, 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., Rosenschein and Zlotkin 1994; Kraus 2001), most results were obtained by means of *simulation* (e.g., Devaux and Paraschiv 2001; Fatima, Wooldridge, and Jennings 2005; Henderson et al. 2008). Devaux and Paraschiv (2001) present work that compares agents negotiating in the Internet agent-based markets. In particular, they compare a strategy of their own agent with behavioral-based strategies taken from the literature (Faratin, Sierra, and Jennings 1998). The simulations are performed in an abstract domain where agents need to negotiate the price of a product. Similarly, Henderson et al. (2008) present results of the performance of various negotiation strategies in a simulated car hire scenario. Finally, Matos, Sierra, and Jennings (1998) 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 of 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 makes those simulations limited. A similar argument to this end has been put forward in Hindriks and Tykhonov (2008b). 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.

Manistersky, Lin, and Kraus (2008) 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 may be somewhat related to the main focus of our paper are negotiation support systems (e.g., the Interactive Computer-Assisted Negotiation Support system (ICANS) presented in Thiessen, Loucks, and Stedinger (1998), the InterNeg Support Program for Intercultural REsearch (INSPIRE)), however, GENIUS advances the state-of-the-art by also providing evaluation mechanisms that allow a quick and simple evaluation of strategies and the facilitation of automated negotiator's design. INSPIRE, by Kersten and Noronha (1999), is a Web-based negotiation support system with the primary goal of facilitating negotiation research in an international setting. The system enables negotiation between two people, 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, unlike GENIUS, 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 (Hales 2002) is more similar to GENIUS than all the other support systems. The Neg-o-Net model is a generic agent-based computational simulation model for capturing multiagency negotiations concerning resource and environmental management decisions. The Neg-o-Net model includes both a negotiation algorithm and some agent models. An 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 the agent's script to model his/her preferences with regard to states and actions. Although Neg-o-Net is much similar to GENIUS, it has two downsizes. First, they currently do not support the incorporation of human negotiators, but only automated ones. Second, they do not provide any evaluation mechanism of the strategies as GENIUS provides.

3. THE GENIUS SYSTEM

The aim of the environment, that is GENIUS, is to facilitate the design of negotiation strategies. Using GENIUS, programmers can focus mainly on the strategy design. This is achieved by GENIUS providing both a flexible and easy-to-use environment for implementing agents and mechanisms that support the strategy design and analysis of the agents. Moreover, the core of GENIUS can be incorporated in a larger negotiation support system that is able to fully support the entire negotiation from beginning to end (examples include the Pocket Negotiator, Hindriks and Jonker (2008), and an animated mediator, Lin, Gev, and Kraus (2011)).

GENIUS is focused on *bilateral* negotiation, i.e., a negotiation between two parties or agents A and B. The agents negotiate over *issues* that are part of a negotiation *domain*, and every issue has an associated range of alternatives or *values*. A negotiation outcome consists of a mapping of every issue to a value, and the set Ω of all possible outcomes is called the *outcome space*. The outcome space is common knowledge to the negotiating parties and stays fixed during a single negotiation session.

We further assume that both parties have certain preferences prescribed by a *preference profile* over Ω. These preferences can be modeled by means of a normalized utility function U, which maps a possible outcome $\omega \in \Omega$ to a real-valued number in the range [0, 1]. In contrast to the outcome space, the preference profile of the agents is private information.

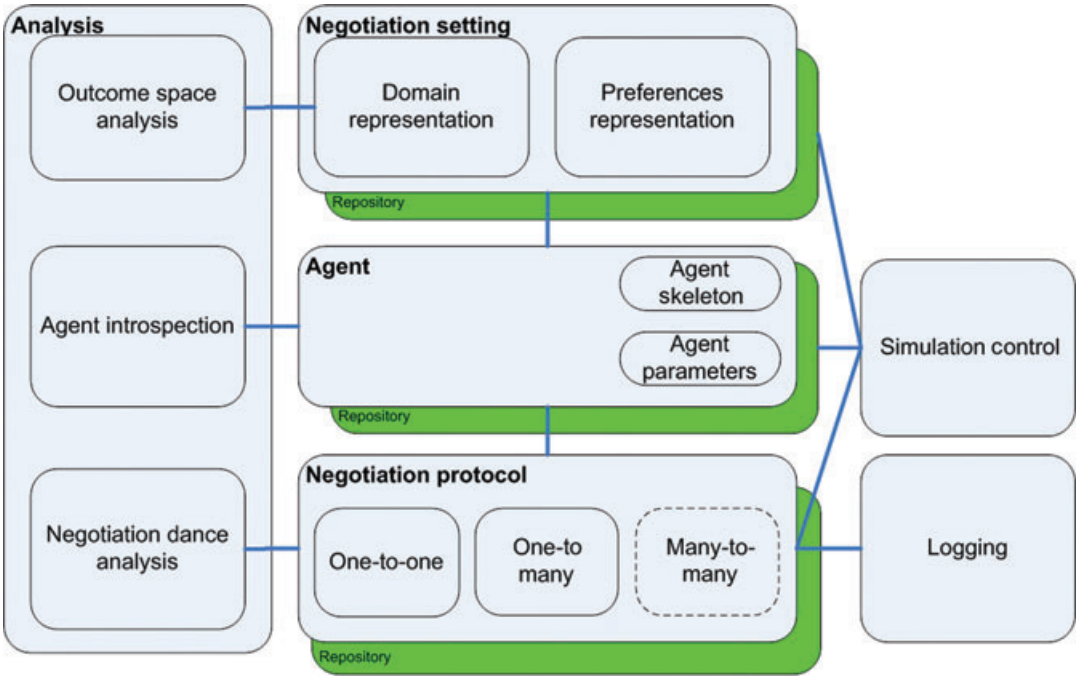

FIGURE 1. GENIUS's architecture.

Finally, the interaction between negotiating parties is regulated by a *negotiation protocol* that defines the rules of how and when proposals can be exchanged.

We begin by describing the detailed and technical architecture of GENIUS and continue with a description of its usage by researchers.

3.1. GENIUS's Architecture

GENIUS provides a flexible simulation environment. Its architecture, presented in Figure 1, is built from several modules: (a) analysis, (b) repository, (c) logging, and (d) simulation control. The analysis module provides researchers the option to analyze the outcomes using different evaluation metrics. The repository contains three different modules of the negotiation that interact with three analysis modules built into GENIUS:

(1) Negotiation scenarios, consisting of a negotiation domain with at least two preference profiles defined on that domain. When a negotiation scenario has been specified, GENIUS is able to perform outcome space analysis on the scenario;
(2) Negotiating agents that implement the Agent Application Programming Interface (API). Agent introspection allows the agents to sense the negotiation environment;
(3) Negotiation protocols, both one-to-one, and multilateral. Depending on the particular protocol, GENIUS can provide negotiation dance analysis to evaluate negotiation characteristics such as fairness, social welfare, and so on.

Finally, the simulation control and logging modules allow researchers to control the simulations, debug it, and obtain detailed information.

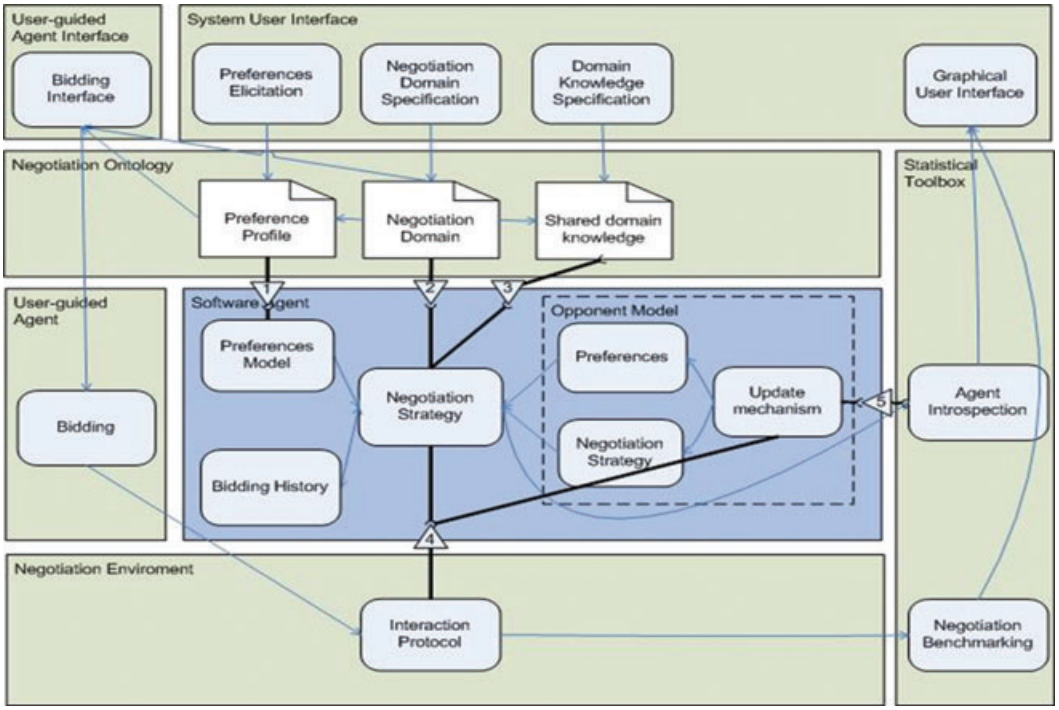

FIGURE 2. GENIUS's detailed architecture.

This is reflected in the detailed architecture presented in Figure 2. The top of the diagram represents the User Interface Layer, in which the user can specify its preferences during the preparation and exploration phase of the negotiation. The Negotiation Ontology Layer below helps to represent the negotiation domain and preference profiles. Currently, linear utility functions are used to model preferences but additional representations can be integrated into the system. The negotiation ontology can be accessed by the Agent Layer to retrieve all relevant negotiation factors pertaining to a particular negotiation scenario. Finally, at the bottom of the diagram, the Negotiation Environment Layer defines the interaction protocol between the agents.

3.2. GENIUS as a Tool for Researchers

GENIUS enables negotiation between automated agents, as well as people. In this section we describe the use of GENIUS before the negotiation, during the negotiation, and afterward.

3.2.1. Preparation Phase. 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 GENIUS by means of adapters.[1]

[1] Indeed as was shown in Hindriks, Jonker, and Tykhonov (2008).

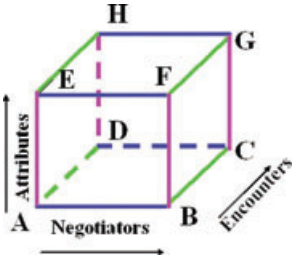

FIGURE 3. Variations of the negotiation settings.

When designing an automated agent, the designer needs to take into account the settings in which the agent will operate. The setting determines several parameters which dictate the number of negotiators taking part in the negotiation, the time frame of the negotiation, and the issues on which the negotiation is being conducted. The negotiation setting also consists of a set of objectives and issues to be resolved. Various types of issues can be involved, including discrete enumerated value sets, integer-value sets, and real-value sets. The negotiation setting can consist of noncooperative or cooperative negotiators. Generally speaking, cooperative agents try to maximize their combined joint utilities (e.g., see Zhang, Lesser, and Podorozhny 2005) whereas noncooperative agents try to maximize their own utilities regardless of the other sides' utilities. Finally, the negotiation protocol defines the formal interaction between the negotiators: whether the negotiation is done only once (one shot) or repeatedly, and how the exchange of offers between the agents is conducted. In addition, the protocol states whether agreements are enforceable or not, and whether the negotiation has a finite or infinite horizon. The negotiation is said to have a finite horizon if the length of every possible history of the negotiation is finite. In this respect, time costs may also be assigned and they may increase or decrease the utility of the negotiator.

Figure 3 depicts the different variations in the settings. GENIUS provides a test bed which allows the designer to easily vary and change these negotiation parameters.

Using GENIUS, a researcher can setup a single negotiation session or a tournament via the graphical user interface (GUI) simulation (see Figure 4) using the negotiation domains and preference profiles from a repository (top left corner of the GUI simulation), and choose strategies for the negotiating parties (bottom left corner of the GUI simulation, as indicated by the "Agents" label). For this purpose, a GUI layer provides options to create a negotiation domain and define agent preferences. This also includes defining different preferences for each role.

A preference profile specifies the preferences regarding possible outcomes of an agent. This can be considered a mapping function that maps the 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. Figure 5 demonstrates how a preference profile can be modified using GENIUS.

Seven negotiation domains are currently available 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 Raiffa (1982), and has over 7,000,000 possible agreements. Issues in the repository have different predictabilities of the evaluation

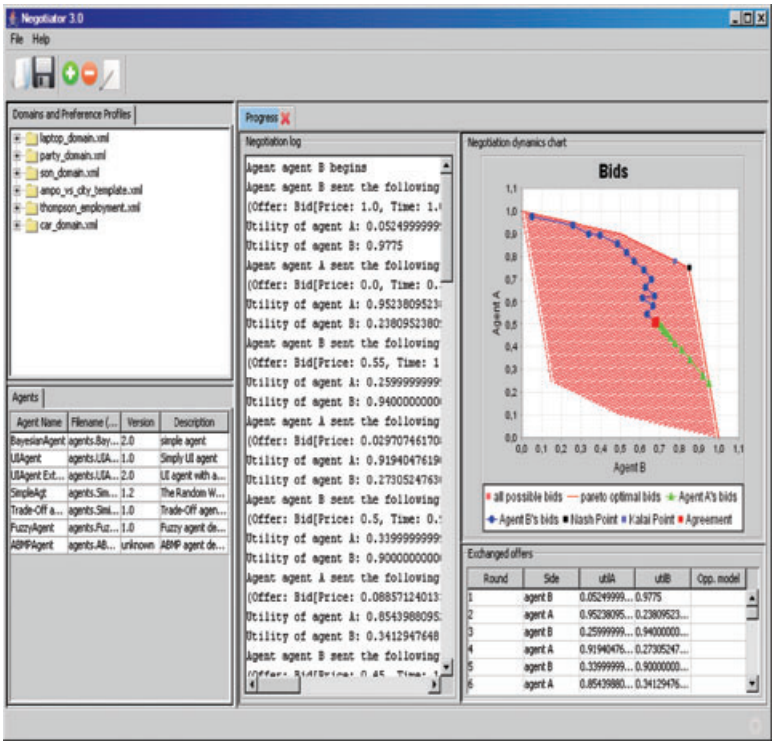

FIGURE 4. An example of GENIUS's main user interface, showing the results of a specific negotiation session.

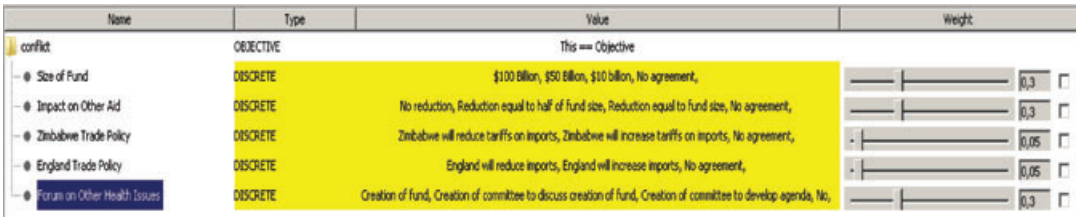

FIGURE 5. Setting the preference profile.

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 Hindriks, Jonker, and Tykhonov 2007; Hindriks and Tykhonov 2008b). The repository of strategies currently contains six automated negotiation strategies, such as the ABMP strategy by Jonker and Treur (2001), the Zero-Intelligence strategy by Hindriks et al. (2007), the QO-strategy by Lin et al. (2008), the Bayesian strategy by Hindriks and Tykhonov (2008a), and others. The repositories of domains and of agents allow agent designers to test their agents on the different domains and against different kinds of agents and strategies.

3.2.2. Negotiation Phase. Human negotiators and automated ones can be joined in a single negotiation session. Human negotiators interact with GENIUS via a GUI. GUIs included in GENIUS allow the human negotiator to exchange offers with his/her counterpart,

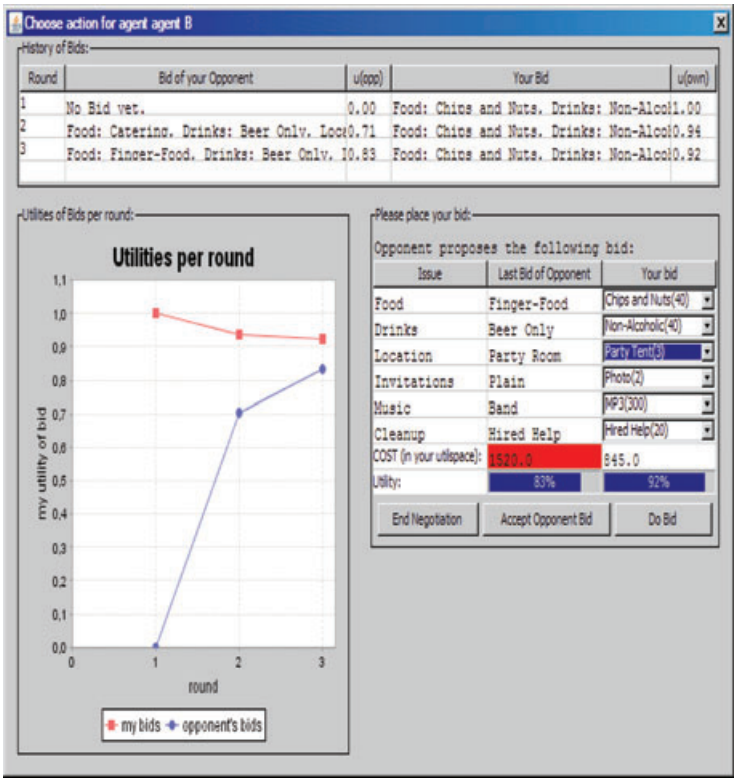

FIGURE 6. An example of the GUI interface of GENIUS for human negotiators during a specific negotiation session.

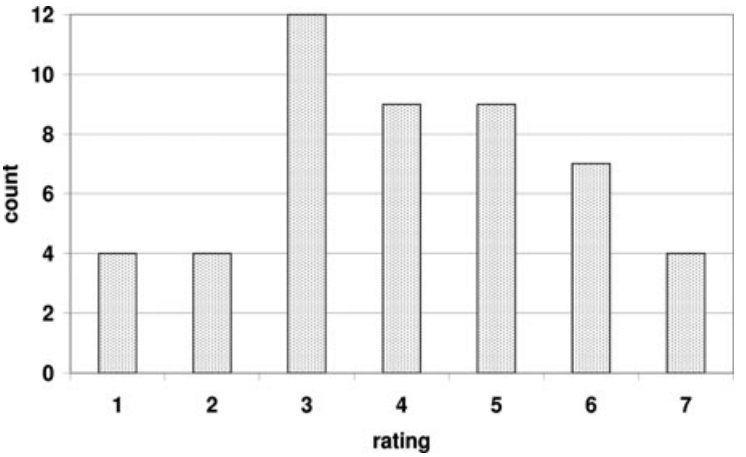

FIGURE 7. Rating of the helpfulness of the analytical toolbox.

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 6 depicts an example of a human negotiator GUI, whereas Figure 8 presents GENIUS's GUI of a tournament session, which allows several agents to be matched against each others.

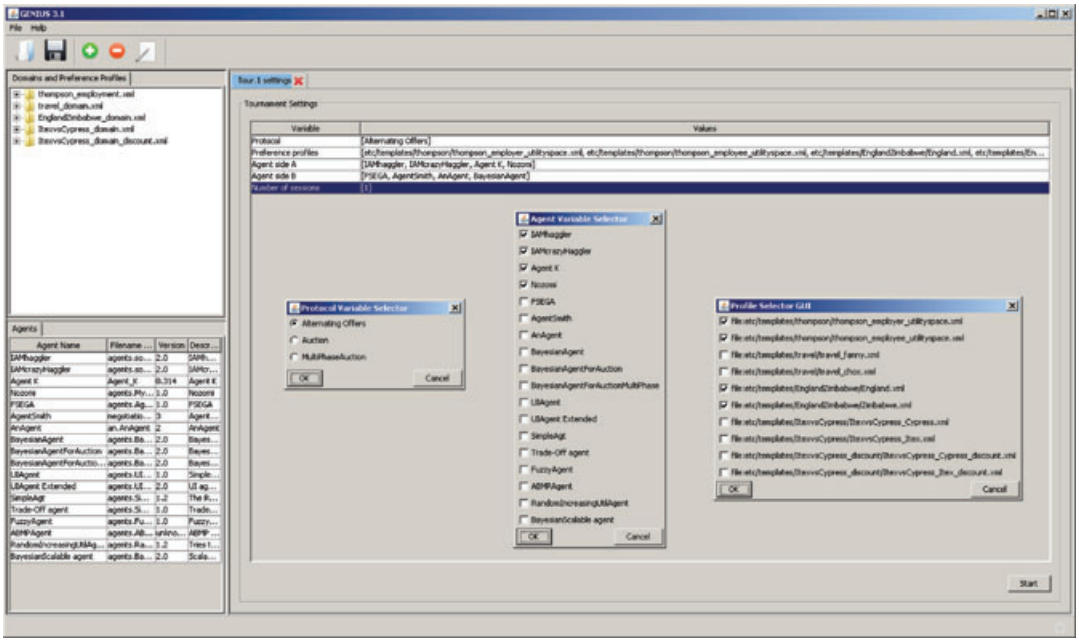

FIGURE 8. Setting up a tournament session for ANAC 2010 involved choosing a protocol, the participating agents, and appropriate preference profiles.

3.2.3. Postnegotiation Phase. GENIUS provides an analytical toolbox for evaluating negotiation strategies (as shown in the Statistical Toolbox module in the right corner of Figure 2). This allows to review the performance and benchmark results of negotiators (whether humans or automated) that negotiated using the system. The toolbox calculates optimal solutions, such as the Pareto efficient frontier, Nash product, and Kalai–Smorodinsky (Raiffa 1982). These solutions are visually shown to the negotiator or the designer of the automated agent, as depicted in the top right corner of Figure 4. 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 own offers from this Pareto frontier as well as the distance from previous offers (as shown by the two lines inside the curve). Also, the designer can inspect both agents' proposals using the analytical toolbox. We note, however, that the visualization of the outcome space together with the Pareto frontier is only possible when we have complete information of both negotiating parties, i.e., both negotiating parties have been assigned a preference profile. In particular, the agent themselves are not aware of the opponent utility of bids in the outcome space and do not know the location of the Pareto frontier. The researcher however, is presented the external overview provided by GENIUS that combines the information of both negotiation parties.

Using the analytical toolbox one can analyze the dynamic properties of a negotiation session, such as a classification of negotiation moves (a stepwise analysis of moves) and the sensitivity to a counterpart's preferences measure, as suggested by Hindriks et al. (2007). 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.

Moreover, negotiating agents designed using heuristic approaches need extensive evaluation, typically through simulations and empirical analysis, as it is usually hard to predict precisely how the system and the constituent agents will behave in a wide variety of circumstances. To do so, there is a genuine need for the development of a best practice repository for negotiation techniques. That is, a coherent resource that describes which negotiation techniques are best suited to a given type of problem or domain. Repositories of agents and negotiation domains available in GENIUS make it an attractive tool for test bedding negotiating agents. To steer the research in the direction of negotiating agents an ANAC was organized[2] using the GENIUS environment, as described in Section 5.

4. EXPERIMENTS

The experiments described below were conducted to test the efficacy of the mechanisms incorporated in GENIUS. 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 to validate the results. These experiment results show that GENIUS allows creating efficient 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

Although 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 toward 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. The incomplete information is expressed as uncertainty regarding the utility preferences of the opponent regarding the. That is, the different possible utility functions of the counterpart were public knowledge, but its exact utility function was unknown.

The first two domains are taken from Lin et al. (2008), 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.

[2] For more details on the ANAC competition see: http://mmi.tudelft.nl/anac.

TABLE 1. Varying Variables in the Experiments.

Experiment	Independent variable	Dependent variable
1	Availability of the analytical toolbox	Negotiator's strategy
2	Availability of additional domains and agents	

The first domain involved reaching an agreement between Britain and Zimbabwe 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." In this domain, four different attributes are under negotiation, resulting in a total of 576 possible agreements.

In the second domain, 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 this scenario, five different attributes are negotiable with a total of 1,296 possible agreements.

The last domain involves finalizing a project plan between Bob and Alice. 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. 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. To validate the efficacy of the two different mechanisms available in GENIUS—the analytical toolbox and the repositories of domains and agents—after each tournament, the students were exposed to only one of these mechanisms and were allowed to redesign 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, as summarized in Table 1. In the first, we evaluated the efficacy of the analytical toolbox. The second experiment was designed to enable evaluation of the efficacy of the domain and agent repositories. We describe both experiments in the following subsections.

The experiments involved bilateral negotiation and were based on the alternating offer protocols, in which offers are exchanged in turns (Rubinstein 1982). All domains had a finite horizon, that is, the length of every possible history of the negotiation is finite with incomplete information about the preferences of the counterpart. The negotiation involved of a finite set of multiattribute issues and time constraints, depending on the domain. During the negotiation process, the agents might gain or lose utility over time. If no agreement is reached by the given deadline a status quo outcome is enforced. The negotiation can also end if one of the parties opts out, thus forcing the termination of the negotiation. When running the automated agents in the tournament, we assigned to each agent a specific utility function, such that in each tournament a different utility function was used.

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 aim of this experiment was to evaluate the efficacy of the evaluation metrics embedded in the analytical toolbox as a key for improving negotiators' strategies. The independent variable in this experiment was the availability (or lack of it) of the analytical toolbox, although the dependent variable was the automated agent designed by the students.

The experiment was conducted as follows. The students were instructed to design an automated negotiator which would be able to negotiate in several domains, however, they were only given the Job Candidate domain described in Section 4.1 as an example. In addition, three automated negotiators were supplied with the tool:[3]

(1) An agent that follows the Bayesian strategy (Hindriks and Tykhonov 2008a);
(2) Another automated agent that follows the ABMP strategy, which is a concession-oriented negotiation strategy (Jonker and Treur 2001), 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. Though, the domain involved incomplete information, they were explained that the visualization is based on complete information and they can evaluate it each time based on a specific preference profile of their counterpart. Then they were allocated several days in which they could use it to redesign 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 Britain–Zimbabwe domain, of which they were unaware during the design process. This allowed us to evaluate whether the analytical toolbox itself is sufficient for designing generic agents.

4.2.2. Evaluation of the Domain and Agent Repositories. In this experiment, as in the previous experiment, 14 graduate students were required to design an automated negotiator using the GENIUS environment. The aim of this experiment was to evaluate the efficacy of existence of domains for generating efficient domain-independent negotiation's strategies. The independent variable in this experiment was the availability (or lack of it) of several domains, although the dependent variable, as in the previous experiment, was the automated agent designed by the students.

The students were aware of the fact that their agent 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

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

TABLE 2. Average Utility Values Gained by the Automated Agents Before and After Being Exposed to the Analytical Toolbox.

Approach/role	Employer	Job candidate
Original agents	517	490
Revised agents	525	505

an example. After their submissions, they were given an additional domain, the Britain–Zimbabwe domain described in Section 4.1. As in the previous experiment, they were allocated several days in which they could redesign their agents based on the newly 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 Britain–Zimbabwe domain of which they were unaware at the time. The second time, each revised agent was matched against all 14 revised agents in the Job Candidate domain and in the Britain–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. Finally, we ran the students' agents against each other using a new domain, the Class Project domain, of which the designers were unaware 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 sufficient for designing efficient generic 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 efficient general automated negotiators.

As we mentioned earlier, we experimented in three distinct domains. The utility values ranged from −575 to 895 for the Britain 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 evaluated the design of the agents using both quantitative results and qualitative results. The quantitative results, presented in Table 2, comprise a comparison of the agents' performance in the different settings of the experiments, although the qualitative results were gathered from the questionnaires the subjects filled in after the submission of the revised agents.

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 compared to the average utilities of the original agents (517 and 490, respectively).

To assess the ease of use of the GENIUS environment in creating generic agents, as well as the helpfulness of the analytical toolbox, the students were asked to answer several questions on a questionnaire they were administered. Note that, 67% of the students indicated that they

TABLE 3. Average Utility Values Gained by the Automated Agents before and after Being Exposed to Past Negotiation Logs.

Approach/role	Employer	Job candidate
Original agents	363	336.8
Revised agents	384.29	365.78

TABLE 4. Average Utility Values Gained by the Automated Agents before and after Being Exposed to an Additional Domain.

Britain–Zimbabwe domain		
Approach/role	Britain	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

redesigned 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, on 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 at 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 7 presents the total rating the students gave for the helpfulness of the analytical toolbox.

Although this encouraged us, as to the efficacy of the analytical toolbox as a supporting mechanism for designing automated negotiators, we still needed to verify whether it could also assist in the design of generic automated negotiators. To test the generality of the agents, we ran the revised agents in a new domain, the Britain–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, compared to almost double the amount of negotiations completed successfully on the known domain (64.4%). That is, although the analytical toolbox was indeed helpful to the students and assisted them in the design of their agent, it was not sufficient for them to design an efficient generic 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.

4.3.2. Experiments with Domain and Agent Repositories. We continued to test other aspects of GENIUS to see whether they help in the design process of agents' strategies. The results are summarized in Tables 3 and 4. In the first part, the students were required to design a generic 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. 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 agent repositories 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 noted, thanks to the logs, that they had bugs in their strategy or that their agents' behavior was too strict and less compromising, leading to excessive negotiations which ended in opting out. Using this insight, they revised their agents' behavior.

To evaluate the benefits of the domain repositories to the performance of their agent, we first matched the students' original agents against each other in the new Britain–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 Britain role and -413.57 for the Zimbabwe role. The results of the revised agents were significantly better in the case of Britain (an average utility of 369.99 with a p-value < 0.03), although 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 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, 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 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 student-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 Britain–Zimbabwe domain.

In this set of experiments we also gave the students questionnaires to help qualitatively assess the efficiency of the domain and agent repositories. The students had to rate several statements on a scale of 1 (being the lowest) to 7 (being the highest). The students indicated that their agents were 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. THE FIRST ANAC

In May 2010, we organized the first ANAC (see Baarslag et al. 2010) in conjunction with the Ninth International Conference on Autonomous Agents and Multiagent

TABLE 5. The Seven Participating Teams of ANAC 2010 from Five Different Universities.

IAMhaggler	University of Southampton
IAMcrazyHaggler	University of Southampton
Agent K	Nagoya Institute of Technology
Nozomi	Nagoya Institute of Technology
FSEGA	Babes Bolyai University
Agent Smith	TU Delft
Yushu	University of Massachusetts Amherst

TABLE 6. The Three Domains Used at ANAC 2010.

	Itex–Cypress	Zimbabwe–Britain	Travel
Number of issues	4	5	7
Size	180	576	188,160
Opposition	Strong	Medium	Weak

Systems (AAMAS-10). In this section, we analyze the benefits of using GENIUS for this competition.

Seven teams from five different universities participated in ANAC 2010, as listed in Table 5. Each team had to design and build a negotiation agent using the GENIUS framework.

We selected three domains and profiles on which the negotiating agents had to reach an agreement. We aimed for a good spread of domain parameters, such as the number of issues, the number of possible proposals, and the opposition of the domain (see Table 6).

In one scenario, taken from Kersten and Zhang (2003), a buyer–seller business negotiation is held between Itex Manufacturing, a producer of bicycle components and Cypress Cycles, a builder of bicycles. The Zimbabwe–Britain domain (Lin et al. 2008) is the same we used for the experiments in which we tested the efficacy of GENIUS (see Section 4.1). Finally, a travel domain was used at ANAC 2010, whereby two friends negotiate on the location of their next holiday. All of the domains were constructed using GENIUS environment.

We refer the reader to Baarslag et al. (2010) for more details on the tournament setup and results of the negotiation competition. We now proceed with a description of the benefits of GENIUS for this competition.

5.1. GENIUS—An Agent Development Tool

As we mentioned in Section 3, the GENIUS framework provides skeleton classes to facilitate the design of negotiating agents. Other aspects of negotiation—specifying information about the domain and preferences, sending messages between the negotiators while obeying a specified negotiation protocol, declaring an agreement—is handled by the negotiation environment. This allows the agent's designer to focus on the implementation of the agent. The agent's designer only needs to implement an agent interface provided by the GENIUS framework. In essence, the agent's developer implements two methods: one for receiving a proposal, and one for making a proposal. The rest of the interaction between the agents is controlled by GENIUS.

TABLE 7. Highlighted Functionality of the API Available to the Agent to Access Information about the Negotiation Environment and Its Preferences.

Agent
Action chooseAction() Enables the agent to offer a bid to the opponent. String getName() Returns the name of the agent. List<AgentParam> getParameters() Accesses any parameters that are externally given to the agent. Timeline getTimeline() Gets information about possible time constraints. Double getUtility(Bid bid) Computes the *discounted* utility of a bid, given the current Timeline. UtilitySpace getUtilitySpace() Gets the preference profile of the agent. receiveMessage(Action opponentAction) Informs the agent about the opponent's action.
Bid
Value getValue(Issue issue) Returns the selected value of a given issue in the current bid. setValue(Issue issue, Value value) Sets the value of an issue.
Domain
List<Issue> getIssues() Gets all issues of the negotiation domain. long getNumberOfPossibleBids() Returns the total amounts of bids the agents can make.
IssueDiscrete (implements Issue)
String getDescription() Returns a short description of the issue. String getName() Returns the name of the issue. List<ValueDiscrete> getValues() Returns all values associated with this issue.
Timeline
Double getElapsedSeconds() Returns the seconds that have elapsed since the start of the negotiation. Double getTime() Gets the normalized elapsed time in [0, 1]. long getTotalSeconds() Gets the total negotiation time in seconds.
UtilitySpace
Double getDiscountFactor() Gets the discount factor. Bid getMaxUtilityBid() Computes the best possible bid given the preferences of the agent. Double getReservationValue() Gets the agent's reservation value. Double getUtility(Bid bid) Computes the utility of a given bid.
ValueDiscrete (implements Value)
String getValue() Returns the text representation of this value.

GENIUS was freely available to the ANAC participants and researchers to develop and test their agent. Table 7 gives an overview of the most important information that was available to the agent through the API provided by GENIUS.

5.2. GENIUS—A Tournament Platform and Analysis Tool

The flexibility provided by the built-in general repository makes GENIUS an effective tournament platform. The contestants of ANAC are able to upload their agent source code

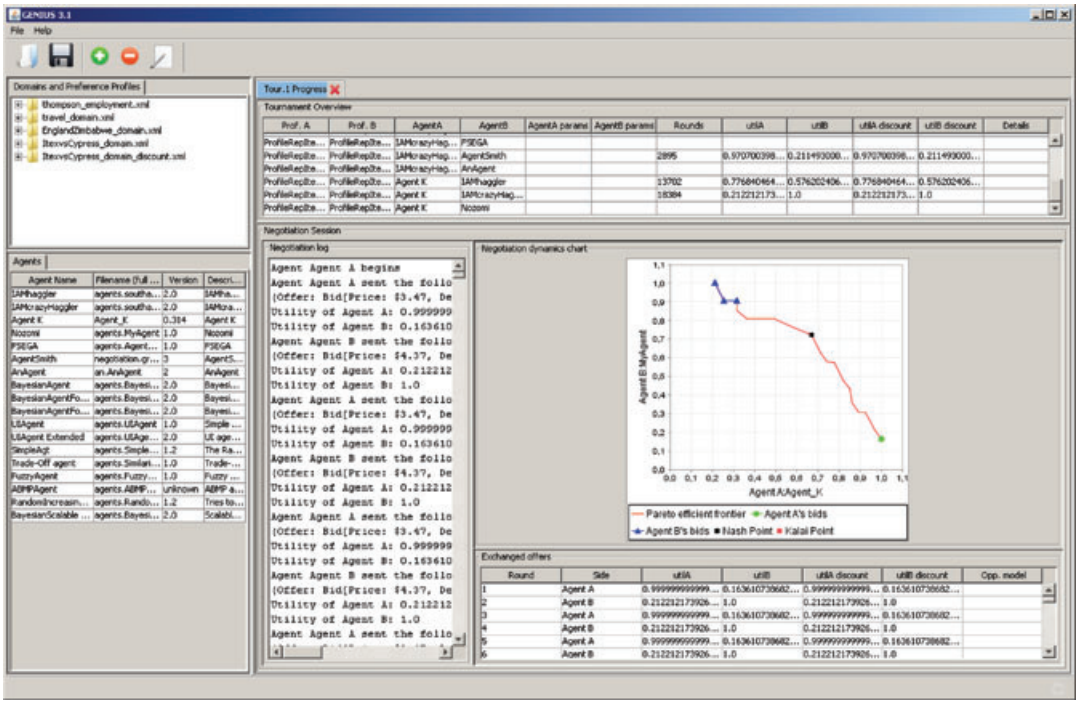

FIGURE 9. A tournament session with the ANAC 2010 agents using the GENIUS interface.

(or even compiled code) to the ANAC organizers. The agents are then added to the GENIUS repository. The ANAC agents and domains are bundled in the GENIUS repository and released to the public after the tournament. GENIUS also provides a uniform, standardized negotiation protocol and scoring system, as every developing team implements the agent inside the same GENIUS environment.

GENIUS supports a number of different protocols, such as the alternating offers protocol, one-to-many auctions, and many-to-many auctions. See Figure 8 for an overview of the types of tournaments that can be run.

The analytical toolbox of GENIUS (see Figure 9) provides a method to evaluate the negotiation strategies employed by the ANAC participants. The toolbox gives valuable graphical information during the negotiation sessions, including: Pareto optimal solutions, Nash product, Kalai–Smorodinsky. The negotiation log gives insight into the agent's reasoning process and can help improve the agent code. When a particular negotiation has finished, an entry is added to the tournament overview, containing information about the number of rounds used, and both utilities associated with the agreement that was reached by the agents. This information can be used to assess the optimality of the agreements reached, either for both agents, or for each agent individually. The result of the analysis can help new agent designers to improve their agents as they play against previous ANAC strategies.

For example, when analyzing the tournament, we have observed that *Agent K* has won by a relatively large margin, yet it only dominated the Travel domain. On Itex–Cypress and Britain–Zimbabwe domains, it earned second place after *Nozomi* and *Yushu*, respectively. However, *Agent K* has won the competition due to its consistent high scores in all domains. Most of the agents had problems on the Travel domain, the biggest domain of ANAC 2010. With such a large domain it becomes unfeasible to enumerate all possible proposals. Only

Agent K, *Nozomi*, and *IAM(crazy)Haggler* were able to effectively negotiate with each other in this domain, which resulted in less break-offs for them, hence their higher scores.

5.3. New Developments in GENIUS

At the end of ANAC 2010, the participating teams held a closing discussion. The consensus among participants was that ANAC was a success, and that the basic structure of the game should be retained. The discussion also yielded valuable suggestions for improving the design of GENIUS for future ANAC competitions:

(1) Domains with discount factors should be included in the tournament.
(2) Changes should be made to the deadline setup.

The generic setup of GENIUS makes it easy to extend it for use with new protocols and negotiation rules. We released a new, public build of GENIUS[4] containing all relevant aspects of ANAC 2010. In particular, this includes all domains, preference profiles, and agents that were used in the competition, in addition to the proposed improvements that were decided upon during the discussion. Consequently, the complete setup of ANAC is available to the negotiation research community.

6. CONCLUSIONS

This paper presents a simulation environment that supports the design of generic 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 important as real-life negotiations are typically differentiated from one another. Furthermore, developing a good domain-dedicated strategy takes weeks and requires talent to do so.

We conducted experiments with automated agents in three distinct domains. The largest domain comprised more than 15,000 possible agreements. Although 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.

We conducted the first ANAC based on the GENIUS environment. GENIUS is freely available for participants to develop and test their agent. Its easy to use agent skeleton makes it a suitable platform for negotiating agent development. GENIUS has the ability to run a wide range of different tournaments, an extensive repository of different agents and domains, and it contains standardized protocols and a scoring system, thus making it the perfect tournament platform for ANAC.

Finally, GENIUS has proved itself as a valuable and extendable research and analysis tool for (post) tournament analysis. ANAC already yielded new state-of-the-art negotiation strategies. Moreover, in light of the analysis of the results, we expect that next year even more sophisticated negotiation strategies will be developed. The second ANAC took place using

[4] http://mmi.tudelft.nl/negotiation/index.php/Genius

a new GENIUS version in conjunction with the AAMAS conference in 2011. The success of the first ANAC underlines the importance of a flexible and versatile negotiation simulation environment such as GENIUS.

The development of GENIUS was crucial to the organization of ANAC and conversely, ANAC also has had an impact on the evolution of GENIUS. GENIUS is constantly being improved. We have released a new, public build of GENIUS that contains all relevant components of ANAC 2010, to make the complete setup of ANAC available to the negotiation research community. We also plan to continue and extend the capabilities of GENIUS, including the incorporation of nonlinear utility functions.

Future research will enable the use of GENIUS for the design of automated negotiators that can successfully and efficiently negotiate with human negotiators. We observed that 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.

We plan to run complete tournaments between the agents in the repository on all available negotiation domains. This will 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 will 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.

We also plan to use GENIUS as a training environment to teach people negotiation concepts, such as exploration of outcome spaces, analysis of opponent's offers, trade-offs between issues, using concession tactics, etc. Another research direction includes the extension of GENIUS to enable argumentation and explanation, by allowing the agents to explain their actions to people.

ACKNOWLEDGMENTS

This research is based upon work supported in part under NSF grant 0705587, by the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF-08-1-0144 and by ERC grant #267523. Furthermore, this research is supported by the Dutch Technology Foundation STW, applied science division of NWO, and the Technology Program of the Ministry of Economic Affairs. It is part of the Pocket Negotiator project with grant number VIVI-project 08075.

REFERENCES

BAARSLAG, T., K. HINDRIKS, C. M. JONKER, S. KRAUS, and R. LIN. 2010. The first automated negotiating agents competition (ANAC 2010). *In* New Trends in Agent-Based Complex Automated Negotiations. *Edited by* T. Ito, M. Zhang, V. Robu, S. Fatima, and T. Matsuo, Volume 383 of Series on Studies of Computational Intelligence. Springer-Verlag: Berlin Heidelberg, pp. 113–135.

BAZERMAN, M. H., and M. A. NEALE. 1992. Negotiator rationality and negotiator cognition: The interactive roles of prescriptive and descriptive research. *In* Negotiation Analysis. *Edited by* H. P. YOUNG. The University of Michigan Press: Ann Arbor, pp. 109–130.

DEVAUX, L., and C. PARASCHIV. 2001. Bargaining on an internet agent-based market: Behavioral vs. optimizing agents. Electronic Commerce Research, **1**:371–401.

FARATIN, P., C. SIERRA, and N. R. JENNINGS. 1998. Negotiation decision functions for autonomous agents. International Journal of Robotics and Autonomous Systems, **24**(3–4):159–182.

FARATIN, P., C. SIERRA, and N. R. JENNINGS. 2002. Using similarity criteria to make (Q11a) issue trade-offs in automated negotiations. Artificial Intelligence Journal, **142**(2):205–237.

FATIMA, S. S., M. WOOLDRIDGE, and N. R. JENNINGS. 2005. A comparative study of game theoretic and evolutionary models of bargaining for software agents. Artificial Intelligence Review, **23**(2):187–205.

FICICI, S., and A. PFEFFER. 2008. Modeling how humans reason about others with partial information. *In* Proceedings of AAMAS'08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 315–322.

HALES, D. 2002. Neg-o-net - a negotiation simulation test-bed. Technical Report CPM-02-109, CPM.

HENDERSON, P., S. CROUCH, R. J. WALTERS, and Q. NI. 2003. Comparison of some negotiation algorithms using a tournament-based approach. *In* Agent Technologies, Infrastructures, Tools, and Applications for E-Services. *Edited by* R. Kowalcyzk, J. P. Muller, H. Tianfield and R. Unland. Springer-Verlag: Berlin Heidelberg, pp. 137–150.

HINDRIKS, K., C. JONKER, and D. TYKHONOV. 2007. Negotiation dynamics: Analysis, concession tactics, and outcomes. *In* Proceedings of IAT'07. IEEE Computer Society: Washington, DC, pp. 427–433.

HINDRIKS, K., C. JONKER, and D. TYKHONOV. 2010. Towards an open negotiation architecture for heterogeneous agents. *In* 12th International Workshop on Cooperative Information Agents (CIA'08), Volume 5180 of *LNAI*. Springer: Heidelberg Berlin, pp. 264–279.

HINDRIKS, K., and D. TYKHONOV. 2008a. Opponent modelling in automated multi-issue negotiation using Bayesian learning. *In* Proceedings of AAMAS'08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 331–338.

HINDRIKS, K., and D. TYKHONOV. 2010. Towards a quality assessment method for learning preference profiles in negotiation. *In* Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis. *Edited by* W. Ketter, H. La Poutré, N. Sadeh, O. Shehory and W. Walsh. Springer-Verlag: Berlin Heidelberg, pp. 46–59.

HINDRIKS, K. V., and C. M. JONKER. 2008. Creating human-machine synergy in negotiation support systems: Towards the pocket negotiator. *In* Proceedings of HuCom'08. ACM Press: New York, pp. 47–54.

JONKER, C.M., and J. TREUR. 2001. An agent architecture for multi-attribute negotiation. *In* Proceedings of IJCAI'01. Morgan Kaufmann: San Francisco, CA, pp. 1195–1201.

JONKER, C. M., V. ROBU, and J. TREUR. 2007. An agent architecture for multi-attribute negotiation using incomplete preference information. Journal of Autonomous Agents and Multi-Agent Systems, **15**(2): 221–252.

KARP, A. H., R. WU, K. y. CHEN, and A. ZHANG. 2003. A game tree strategy for automated negotiation. Technical Report HPL-2003-154.

KERSTEN, G. E., and S. J. NORONHA. 1999. Www-based negotiation support: Design, implementation, and use. Decision Support Systems, **25**(2):135–154.

KERSTEN, G. E., and G. ZHANG. 2003. Mining inspire data for the determinants of successful internet negotiations. *In* Central European Journal of Operational Research, **11**(3):297–316.

KRAUS, S. 2001. Strategic Negotiation in Multi-Agent Environments. The MIT Press: Cambridge, MA.

KRAUS, S., and D. LEHMANN. 1995. Designing and building a negotiating automated agent. Computational Intelligence, **11**(1):132–171.

KRAUS, S., J. WILKENFELD, M. A. HARRIS, and E. BLAKE. 1992. The hostage crisis simulation. Simulation & Gaming, **23**(4):398–416.

LAX, D. A., and J. K. SEBENIUS. 1992. Thinking coalitionally: Party arithmetic, process opportunism, and strategic sequencing. *In* Negotiation Analysis. *Edited by* H. P. YOUNG. The University of Michigan Press: Ann Arbor, pp. 153–193.

LIN, R., Y. GEV, and S. KRAUS. 2011. Bridging the gap: Face-to-face negotiations with automated mediator. *In* IEEE Intelligent Systems, **26**(6):40–47.

LIN, R., S. KRAUS, J. WILKENFELD, and J. BARRY. 2008. Negotiating with bounded rational agents in environments with incomplete information using an automated agent. Artificial Intelligence Journal, **172**(6–7):823–851.

LOMUSCIO, A., M. WOOLDRIDGE, and N. R. JENNINGS. 2001. A classification scheme for negotiation in electronic commerce. *In* Proceedings of AMEC'01, Springer-Verlag: London, UK, pp. 19–33.

MANISTERSKY, E., R. LIN, and S. KRAUS. 2008. Understanding how people design trading agents over time. *In* Proceedings of AAMAS'08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1593–1596.

MATOS, N., C. SIERRA, and N. R. JENNINGS. 1998. Determining successful negotiation strategies: An evolutionary approach. *In* Proceedings of ICMAS'98. IEEE Computer Society: Washington, DC, p. 182.

RAIFFA, H. 1982. The Art and Science of Negotiation. Belknap Press of Harvard University Press: Cambridge MA.

ROSENSCHEIN, J. S., and G. ZLOTKIN. 1994. Designing Conventions for Automated Negotiation among Computers. The MIT Press: Cambridge.

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

SYCARA, K., and D. ZENG. 1998. Bayesian learning in negotiation. International Journal of Human-Computer Studies - Evolution and Learning in Multiagent Systems, **48**(1):125–141.

THIESSEN, E. M., D. P. LOUCKS, and J. R. STEDINGER. 1998. Computer-assisted negotiations of water resources conflicts. Group Decision and Negotiation, **7**(2):109–129.

WELLMAN, M. P., A. GREENWALD, and P. STONE. 2007. Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition. MIT Press: Cambridge MA.

ZHANG, X.Q., V. LESSER, and R. PODOROZHNY. 2005. Multi-dimensional, multistep negotiation for task allocation in a cooperative system. Journal of Autonomous Agents and Multi-Agent Systems, **10**(1):5–40. Available at http://mas.cs.umass.edu/paper/212. Accessed August 14, 2012.