Decoupling Negotiating Agents to Explore the Space of Negotiation Strategies

Tim Baarslag
Interactive Intelligence Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
T.Baarslag@tudelft.nl

Koen Hindriks
Interactive Intelligence Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
K.V.Hindriks@tudelft.nl

Mark Hendrikx
Interactive Intelligence Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
M.J.C.Hendrikx@tudelft.nl

Alex Dirkzwager
Interactive Intelligence Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
A.Dirkzwager@ii.tudelft.nl

Catholijn M. Jonker
Interactive Intelligence Group
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
C.M.Jonker@tudelft.nl

ABSTRACT

Every year, automated negotiation agents are improving on various domains. However, given a set of automated negotiation agents, current methods allow to determine which strategy is best in terms of utility, but not so much the reason of success. In order to study the performance of the individual components of a negotiation strategy, we introduce an architecture that distinguishes three components which together constitute a negotiation strategy: the bidding strategy, the opponent model, and the acceptance strategy.

Our contribution to the field of bilateral negotiation is twofold: first, we show that existing state-of-the-art agents are compatible with this architecture by re-implementing them in the new framework; secondly, as an application of our architecture, we systematically explore the space of possible strategies by recombining different strategy components, resulting in negotiation strategies that improve upon the current state-of-the-art in automated negotiation.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*intelligent agents, multi-agent systems*

General Terms

Algorithms, Bargaining, Experimentation, Negotiation

Keywords

Automated bilateral negotiation, BOA agent framework, decoupled, component-based, bidding strategy, opponent model, acceptance conditions, acceptance criteria

1. INTRODUCTION

Recently, many new automated negotiation agents have been developed. There is now a large body of negotiation strategies available, and with the emergence of the International Automated Negotiating Agents Competition (ANAC) [2, 4], new strategies are generated on a yearly basis.

While methods exist to determine the best negotiation agent given a set of agents [2, 4], we still do not know which type of agent is most effective in general, and especially why. It is impossible to exhaustively search the large (in fact, infinite) space of negotiation strategies; therefore, there is a need for a systematic way of searching this space for effective candidates.

Many of the sophisticated agent strategies that currently exist are comprised of a fixed set of modules. Generally, a distinction is made between three different modules: one module that decides whether the opponent's bid is acceptable; one that decides which set of bids could be proposed next; and finally, one that tries to guess the opponent's preferences and takes this into account when selecting an offer to send out. The negotiation strategy is a result of the complex interaction between these components, of which the individual performance may vary significantly. For instance, an agent may contain a module that predicts the opponent's preferences very well, but the agent may still perform badly utility-wise because it concedes far too quickly.

This means that overall performance measures, such as average utility obtained in a tournament, make it hard to pinpoint which components of an agent work well. To date no efficient method exists to identify to which of the components the success of a negotiating agent can be attributed. Finding such a method would allow to develop better negotiation strategies, resulting in better agreements; the idea being that well-performing components together will constitute a well-performing agent.

To tackle this problem, we propose to analyze three components of the agent design separately. We show that most of the currently existing negotiating agents can be fitted into the so-called *BOA framework* by putting together three main components in a particular way; namely: the *Bidding strategy*, an *Opponent model*, and an *Acceptance strategy*. We support this claim by re-implementing, among others, the ANAC agents in our framework. Furthermore, we show that the BOA agents are equivalent in behavior and performance to their original counterparts.

The advantages of fitting agents into the BOA framework are threefold: first, it allows to study the behavior and performance of individual components; second, it allows to systematically explore the space of possible negotiation strategies; third, the identification of unique interacting components simplifies the creation of new negotiation strategies.

Finally, we demonstrate the value of our framework by assembling, using already existing components, new negotiating agents that perform better than the current state-of-the-art. This shows that the BOA framework can yield better performing agents by combining better performing components.

The remainder of this paper is organized as follows. Section 2 discusses related work. In Section 3 the BOA agent framework is introduced, and we outline a research agenda on how to employ it. Section 4 provides evidence that many of the currently existing agents fit into the BOA framework, and discusses challenges in decoupling existing negotiation strategies. In Section 4.2 we illustrate how to test for equivalence of the original agent and its decoupled version. Section 5 shows how the BOA framework allows us to combine best practices in current agent design, leading to new, more effective strategies. Finally, in Section 6 we discuss lessons learned and provide directions for future work.

2. RELATED WORK

Since this paper introduces a framework based on a theory of components, we have surveyed literature that investigates and evaluates such components. There are three categories of related work: literature detailing the architecture of the negotiation strategy of an agent; work that discusses and compares the performance of a component of a negotiation strategy; and finally, literature that explores and combines a set of negotiation strategies to find an optimal strategy.

2.1 Achitecture of Negotiation Strategies

To our knowledge, there is little work in literature describing, at a similar level of detail as our work, the generic components of a negotiation strategy architecture. For example, Bartolini et al. [5] and Dumas et al. [8] treat the negotiation strategy as a singular component. There are however some notable exceptions.

Jonker et al. [16] present an agent architecture for multi attribute negotiation, where each component represents a specific process within the behavior of the agent, e.g.: attribute evaluation, bid utility determination, utility planning, and attribute planning. In contrast to our work, Jonker et al. focus on tactics for finding a counter offer and do not discuss acceptance conditions. However, there are some similarities between the two architectures. For example, the utility planning and attribute planning component correspond to the bidding strategy component in our architecture.

Ashri et al. [1] introduce a general architecture for negotiation agents, discussing components that resemble our architecture; however, the negotiation strategy is described from a BDI-agent perspective (in terms of motivation and mental attitudes). Components such as a proposal evaluator and response generator resemble an acceptance condition and bidding strategy respectively.

Hindriks et al. [13] introduce a generic architecture for negotiation agents in combination with a negotiation system architecture. Parts of the agent architecture correspond to the architecture presented in this paper; however, their focus is primarily on how the agent framework can be integrated into a larger system architecture. In addition, Hindriks et al. treat the acceptance condition and bidding strategy as a singular component.

2.2 Components of Negotiation Strategy

Evaluation of the performance of components is important to gain understanding of the performance of a negotiation strategy.

Regarding acceptance conditions, Baarslag et al. [3] analyze the performance of a set of acceptance conditions. These acceptance conditions depend on parameters such as time, utility of previous or next bid, and utility thresholds.

The notion of opponent model as a component of a negotiation strategy has been discussed by various authors, however to our knowledge there is no work comparing the performance of various state-of-the-art opponent models. Recently, Hindriks et al. [15] introduced different quality measures for learning, based on the estimated preference profile and the actual preference profile, but this has not been put to practice yet. Different types of models exist in literature, including opponent models that estimate the reservation value [24], the (partial) preference profile [14], the opponent's acceptance of offers [20], and that predict the opponent's next move [7].

Our work focuses on opponent models which estimate the (partial) preference profile, because most existing implementations fit in this category; however, in principle, our framework can accommodate for modeling the opponent's strategy as well. Our framework also allows to determine and to compare the performance of different opponent models by separating the implementation of the opponent model from the rest of the negotiation agent.

Although we are not the first to identify the BOA components in a negotiation strategy, our approach seems to be unique in the sense that we vary these components of the strategies, thereby creating new negotiation strategies, and improving the state-of-the-art in doing so.

2.3 Negotiation Strategy Space Exploration

There are at least four main types of baseline bidding strategies to compare the performance of a bidding strategy against: time dependent [10, 11], resource dependent [10], behavior dependent [10], and zero intelligence strategies [12].

Faratin et al. [10] start by analyzing the performance of pure negotiation tactics on single issue domains in a bilateral negotiation setting. The decision function of the pure tactic is then treated as a component around which the full strategy is built. While they discuss how tactics can be linearly combined, the performance of linearly combined tactics are not analyzed (in contrast to Matos et al. [19]), as they note that the set of possible strategies is too large to explore.

Matos et al. [19] use a set of baseline negotiation strategies with varying parameters. The negotiation strategies are combined linearly and encoded as chromosomes after which they are utilized by a genetic algorithm to analyze the effectiveness of the strategies. The fitness of an agent is its score in a negotiation competition. This approach is limited to acceptance criteria that specify a utility interval of acceptable values, and hence does not take time into account; furthermore, the agents do not employ explicit opponent modeling.

Eymann [9] also uses genetic algorithms with more complex negotiating strategies, evolving six parameters that influence the bidding strategy. The genetic algorithm uses the current negotiation strategy of the agent and the opponent strategy with highest average income to create a new strategy, similar to other genetic algorithm approaches (see Beam and Segev [6] for a discussion of the application of genetic algorithms in automated negotiation). The genetic algorithm approach mainly treats the negotiation strategy optimization as a search problem in which the parameters of a small set of strategies is varied using genetic algorithms. In our approach, we analyze a more complex space of newly developed negotiation strategies, as our pool of surveyed negotiation strategies consists of strategies introduced in the ANAC competition [2, 4], as well as the strategies discussed by Faratin et al. [10]. Furthermore, each strategy consists of components that can have parameters themselves. Our contribution is to define and implement a framework that allows to easily vary all main components of a negotiating agent.

3. THE BOA AGENT FRAMEWORK

In the last decade, many different negotiation strategies have been introduced in the search for an effective, generic automated negotiator (see related work Section 2). Current work often focuses on optimizing the negotiation strategy as a whole. We propose to direct our attention to a component-based approach, especially now that we have access to a large repository of mutually comparable negotiation strategies due to ANAC. This approach has several advantages:

1. Given measures for the effectiveness of the individual components of a negotiation strategy, we are able to pinpoint the most promising components, which gives insight into the reasons for success of the strategy;

2. Focusing on the most effective components helps to systematically search the space of possible negotiation strategies by recombining them into new strategies.

We make a distinction between two types of components in the sections below: elements that are part of the agent's environment, and components that are part of the agent itself.

3.1 Negotiation Environment

We employ the same *negotiation environment* as in [2, 4, 18]; that is, we consider *bilateral*, *real time* automated negotiations, where the interaction between the two negotiating parties is regulated by the alternating-offers protocol [21]. The agents negotiate over a set of issues, as defined by the negotiation *domain*, which holds the information of possible bids, constraints, and the discount factor. The negotiation happens in real time, and the agents are required to reach an agreement (i.e., one of them has to accept) before the deadline is reached. The timing of acceptance is particularly important because the utility may be discounted, that is: the value of an agreement may decrease over time.

In addition to the domain, both parties also have privately-known preferences described by their *preference profiles*. While the domain is common knowledge, the preference profile of each player is private information. This means that each player only has access to its own utility function, and does not know the preferences of its opponent. The player can attempt to learn this during the negotiation encounter by analyzing the *bidding history*, using an opponent modeling technique.

3.2 The BOA Agent

Based on a survey of literature and the implementations of currently existing negotiation agents, we identified three main components of a general negotiation strategy: a *bidding strategy*, possibly an *opponent model*, and an *acceptance strategy* (BOA). The elements of a BOA agent are visualized in Figure 1. In order to fit an agent into the BOA framework, it should be possible to discern these components in the agent design, with no dependencies between them. An exposition of the agents we considered is given in the next section, which will further motivate the choices made below.

1. **Bidding strategy**. A bidding strategy is a mapping which maps a negotiation trace to a bid. The bidding strategy can interact with the opponent model by consulting with it, passing one or multiple bids and see how they compare within the estimated opponent's utility space.
   Input: *opponent utility of bids, negotiation trace.*
   Output: *provisional upcoming bid.*

2. **Opponent model**. An opponent model is a learning technique that constructs a model of the opponent's negotiation profile. In our approach, the opponent model should be able to estimate the opponent's utility of a given bid.
   Input: *set of possible bids, negotiation trace.*
   Output: *estimated opponent utility of a set of bids.*

3. **Acceptance strategy**. The acceptance strategy determines whether the bid that the opponent presents is acceptable.
   Input: *provisional upcoming bid, negotiation trace.*
   Output: *send accept, or send out the upcoming bid.*

The components interact in the following way (the full process is visualized in Figure 1). When receiving an opponent bid, the BOA agent first updates the *bidding history* and *opponent model* to make sure the most up-to-date data is used, maximizing the information known about the environment and opponent.

Given the opponent bid, the *bidding strategy* determines the counter offer by first generating a set of bids with a similar preference for the agent. The *bidding strategy* uses the *opponent model* (if present) to select a bid from this set by taking the opponent's utility into account.

Finally, the *acceptance strategy* decides whether the opponent's action should be accepted. At first glance, it may seem counter-intuitive to make this decision *at the end* of the agent's deliberation cycle. Clearly, deciding upon acceptance *at the beginning* would have the advantage of not wasting resources on generating an offer that might never be sent out.

However, generating an offer first allows us to employ acceptance conditions that depend on the utility of the counter bid that is ready to be sent out. This method is widely used in existing agents [3]. Such acceptance mechanisms can make a more informed decision by postponing their decision on acceptance until the last step; therefore, and given our aim to incorporate as many agent designs as possible, we adopt this approach in our framework.

If the opponent's bid is not accepted by the acceptance strategy, then the bid generated by the bidding strategy is offered instead.

3.3 Employing the BOA framework

The component-based approach as outlined above enables us to follow at least two approaches: first of all, it allows us to independently analyze the components of every negotiation strategy that fits in to our framework. For example, by re-implementing the ANAC agents in the BOA framework, it becomes possible to compare the accuracy of all ANAC opponent models, and to pinpoint the best opponent model among them. Naturally, this helps to build better agents in the future.

Secondly, we can proceed to *mix* different BOA components, e.g.: replace the opponent model of the runner-up of ANAC by a different opponent model and test whether this makes a difference in placement. Such a procedure enables us to assess the reasons for success of an agent, and makes it possible to systematically search for an effective automated negotiator.

The first part of the approach gives insight in what components are best in isolation; the second part gives us understanding of their influence on the agent as a whole. At the same time, both approaches raise some key theoretical questions, such as:

1. Can the BOA components be identified in all, or at least most, current negotiating agents?

2. How do we measure the performance of the single components? Can a single best component be identified, or does this strongly depend on the other components?

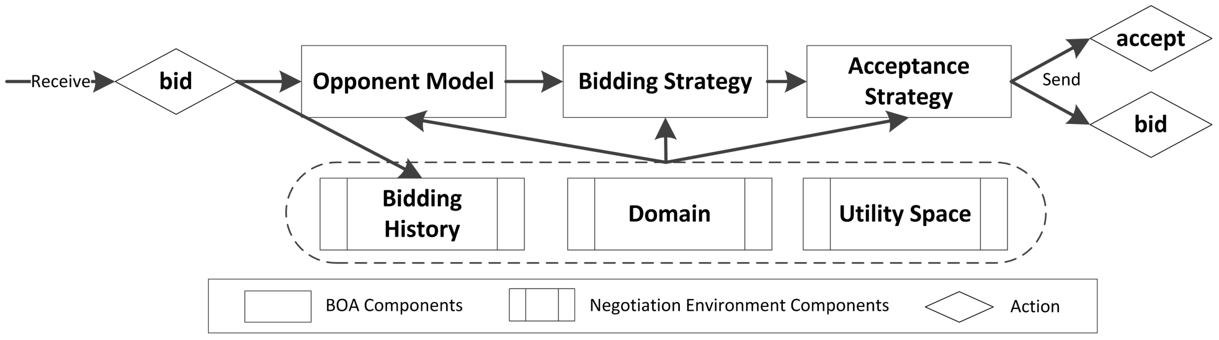

Figure 1: The BOA framework negotiation flow.

3. If the individual components perform better than others (with respect to some performance measure), does combining them in an agent also improve the agent's performance?

In this work we do not aim to fully answer all of the above questions; instead, we outline a research agenda, and introduce the BOA framework as a tool that can be used towards answering these questions.

Nonetheless, in the next section, we will provide empirical support for an affirmative answer to the first theoretical question: indeed, in many cases the components of the BOA framework can be identified in current agents, and we will also provide reasons for when this is not the case.

The answer to the second question depends on the component under consideration: for an opponent model, it is straightforward to measure its effectiveness: the closer the opponent model is to the actual profile of the opponent, the better it is. The quality of approximation can be measured in multiple ways [15], and should be balanced against other measures that also influence its performance. For instance, in a real time negotiation there is a trade-off between the required computational resources and expected quality of the opponent model.

The performance of the other two components of the BOA framework is better measured in terms of utility obtained in negotiation (as has been done for acceptance strategies in [3]), as there seems no clear alternative method to define the effectiveness of the acceptance strategy or bidding strategy in isolation. In any case, the BOA framework can be used as a research tool to help answer such theoretical questions.

Regarding the third question: suppose we take the best performing bidding strategy, equip it with the most faithful opponent model, and combine this with the most effective acceptance strategy; it would seem reasonable to assume this combination results in an effective negotiator. We plan to elaborate on this conjecture in future work (see also Section 6); however, Section 5 will already provide a first step towards this goal by recombining components of the ANAC 2011 agents to create more effective agents than the original versions.

4. DECOUPLING EXISTING AGENTS

In this section we provide empirical evidence that many of the currently existing agents can be decoupled by separating the components of a set of state-of-the-art agents. This section serves three goals: first, we discuss how existing agents can be decoupled in a bidding strategy, acceptance strategy, and possibly an opponent model; second, we argue that the BOA framework design is appropriate, as most agents will turn out to fit in our framework; third, we discuss a method to determine if the sum of the components – the BOA agent – is equal in behavior to the original agent.

4.1 Identifying the Components

In this section we identify the components of seventeen negotiating agents, taken from the ANAC competitions of 2010 [4], and 2011 [2], and of baseline strategies such as the time dependent agents [10, 11], and zero intelligence strategies [12]. We have selected these strategies as they are well-known and/or represent the current state-of-the-art in automated negotiation, having been implemented by various negotiation experts.

Since the agents were not designed with decoupling in mind, all agents had to be re-implemented to be supported by the BOA framework. Our decoupling methodology was to adapt an agent's algorithm to enable it to switch its components, without changing the agent's functionality. A method call to specific functionality, such as code specifying when to accept, was replaced by a more generic call to the acceptance mechanism, which can then be swapped at will. The contract of the generic calls are defined by the expected input and output of every component, as outlined in Section 3.2.

The first step in decoupling an agent is to determine which components can be identified. For example, in the ANAC 2010 agent *FSEGA* [22], an acceptance condition, a bidding strategy, and an opponent model can all be identified. The acceptance strategy combines simple, utility–based criteria (called $\mathbf{AC}_{\text{const}}$ and $\mathbf{AC}_{\text{prev}}$ in [3]) and can be easily decoupled in our framework. The opponent model is a variant of the *Bayesian opponent model* [14, 24], which is used to optimize the opponent utility of a bid. Since this usage is consistent with our framework (i.e., the opponent model provides opponent utility information), the model can be replaced by a generic opponent model interface. The final step is to change the bidding strategy to use the generic opponent model instead of specifically its own model. Other agents can be decoupled using a similar process.

Unfortunately, some agent implementations contained slight dependencies between different components. These dependencies needed to be resolved to separate the design into singular components. For example, both the acceptance strategy and bidding strategy of the ANAC 2011 agent *The Negotiator*[1] rely on a shared target utility. In such cases, the agent can be decoupled by introducing Shared Agent State (SAS) classes. A SAS class avoids code duplication, and therefore performance loss, by containing the code that is shared between the components. One of the components uses the SAS to calculate the values of the required parameters and saves the results, while the other component simply asks for the

[1]Descriptions of all ANAC 2011 agents can be found in [2].

saved results instead of duplicating the calculation.

Table 1 provides an overview of all agents we re-implemented in our framework, and more specifically, which components we were able to decouple. In fact, we were able to decouple all ANAC2011 and ANAC2011 agents except for *ValueModelAgent*. While *ValueModelAgent* can be theoretically decoupled, the strong coupling between its components results in too computationally heavy components when used separately.

As is evident from Table 1, the only possible obstacle in decoupling an agent is its usage of the opponent model. An agent's opponent model can be employed in multiple ways. Some agents, such as *Nice Tit for Tat*, attempt to estimate the Nash point on the Pareto frontier. Other common applications include: ranking a set of bids according to the opponent utility, reciprocating in opponent utility, and extrapolating opponent utility. The generic opponent model interface needs to sufficiently accommodate such requirements from the bidding strategy to make interchangeability possible. For this reason we require the opponent model interface to be able to produce the estimated opponent utility of an arbitrary negotiation outcome.

With regard to the opponent model, there are three groups of agents: first, there are agents such as *FSEGA* [22], which use an opponent model that can be freely interchanged; second, there are agents such as the ANAC 2010 winner *Agent K* [17], which do not have an opponent model themselves, but can be extended to use one. Such agents typically employ a bidding strategy that first decides upon a specific target utility range, and then picks a *random* bid within that range. These agents can easily be fitted with an opponent model instead, by passing the utility range through the opponent model before sending out the bid. Lastly, there are agents that cannot use an opponent model in any meaningful way, such as *Random Walker* [12], and there are agents such as *Gahboninho* and *BRAM Agent* that use a frequency-based opponent model which is not compatible with our framework, as their opponent models do not yield enough information to compute the opponent utility of bids. For this type of agent, we consider the opponent model part of the bidding strategy.

Agent	B	O	A
Agent K	✓	∅	✓
Agent K2	✓	∅	✓
Agent Smith	✓	✓	✓
BRAM Agent	✓	–	✓
FSEGA	✓	✓	✓
Gahboninho	✓	–	✓
HardHeaded	✓	✓	✓
IAMcrazyHaggler	✓	∅	✓
IAMhaggler2010	✓	✓	✓
IAMhaggler2011	✓	∅	✓
Nice Tit for Tat	✓	✓	✓
Nozomi	✓	∅	✓
Offer Decreasing	✓	–	✓
Random Walker	✓	–	✓
TheNegotiator	✓	∅	✓
Time dependent agent	✓	∅	✓
Yushu	✓	∅	✓

Table 1: Overview of components identified in every agent. ✓: original has component that can be decoupled. ∅: original has no component, but it can be added. – : no support for such a component.

When decoupling the agents, it becomes apparent that the bidding strategy component varies greatly between different agents. In contrast, there are only two main types of opponent models being used: Bayesian models and frequency models. Bayesian models are an implementation of a (scalable) model of the opponent preferences that is updated using Bayesian learning [14, 24].

The main characteristic of frequency based models is that they track the frequency of occurrence of issues and values in the opponent's bids and use this information to estimate the opponent's preferences. In practice, Bayesian models are more computationally intensive, whereas frequency models are relatively light-weight.

After comparing the different implementation variants in all agents, we consider the Bayesian model of *IAMhaggler 2010* and the frequency model of *HardHeaded* representative of their type, as we believe that both are the most accurate and computationally efficient implementations.

Similar to the opponent models, most agents use variations and combinations of a small set of acceptance conditions. Specifically, many agents use simple thresholds for deciding when to accept (called $\mathbf{AC}_{\text{const}}$ in [3]) and linear functions that depend on the utility of the bid under consideration ($\mathbf{AC}_{\text{next}}(\alpha, \beta)$ [3]).

4.2 Testing Equivalence of BOA Agents

A BOA agent should behave identically to the agent from which its components are derived. Equivalence can be verified in two ways; first, given the same negotiation environment and the same state, both agents should behave in exactly identical ways (Section 4.2.1); second, the performance in a real time negotiation of both agents should be similar (Section 4.2.2).

4.2.1 Identical Behavior Test

Two deterministic agents can be considered equivalent if they perform the same action given the same negotiation trace. There are two main problems in determining equivalence: first, most agents are nondeterministic, as they behave randomly in certain circumstances; for example, when picking from a set of bids of similar utility; second, the default protocol in GENIUS uses real time [18], which is highly influenced by cpu performance. This entails that in practice, two runs of the same negotiation are never exactly equivalent.

To be able to run an equivalence test despite of the agents choosing random actions, we fixed the seeds of the random functions of the agents. The challenge of working in real time was dealt with by changing the real time deadline to a maximum amount of rounds. Since time does not pass within a round, cpu performance does not play a role.

All agents were evaluated on the ANAC2011 domains (see [2] for a domain analysis). The ANAC2011 domains vary widely in characteristics: the number of issues ranges from 1 to 8, the size from 3 bids to 390.625 bids, and the discount from none (1.0) to strong (0.424). Some ANAC2010 agents, specifically *Agent Smith* and *Yushu*, were not designed for large domains and were therefore run on a subset of the domains.

The *opponent strategies* used in the identical behavior test should satisfy two properties: the opponent strategy should be deterministic, and secondly, the opponent strategy should not be the first to accept, to avoid masking errors in the agent's acceptance strategy. Given these two criteria, we used the standard time-dependent tactics [10, 11] for the opponent bidding strategy. Specifically, we use *Hardliner* ($e = 0$), *Linear Conceder* ($e = 1$), and *Conceder* ($e = 2$). In addition, we use the *Offer Decreasing* agent which offers the set of all possible bids in decreasing order of utility.

All original and BOA agents were evaluated against all four opponents on eight domains, using both preference profiles defined on each domain. An agent running both strategies in parallel was used to check that both strategies were equivalent.

After the experiments were performed, the test results indicated that all BOA agents were exactly identical to their original counterparts.

4.2.2 *Similar Performance Test*

Two agents can perform the same action given the same input, but may still achieve different results because of differences in their real time performance. When decoupling agents, there is a trade-off between performance and interchangeability of components. For example, most agents record only a partial negotiation history, while some acceptance strategies require the full history of the agent and/or its opponent. In such cases, the agent can be constrained to be incompatible with these acceptance strategies, or generalized to work with the full set of available acceptance strategies. We typically elected the most universal approach, even when this negatively influenced performance. We will demonstrate that while there is some performance loss when decoupling existing agents, it does not significantly impact the negotiation outcome.

The performance of the BOA agents was tested by letting them participate in the ANAC 2011 tournament (using the same setup, cf. [2]). The decoupled ANAC 2011 agents replaced the original agents, resulting in an 8×8 tournament, while the ANAC 2010 agents were added to the tournament, resulting in 9×9 tournaments.

For our experimental setup we used computers that were slower compared to the IRIDIS high-performance computing cluster that was used to run ANAC 2011. As we were therefore unable to reproduce exactly the same data, we first recreated our own ANAC 2011 tournament data (Appendix B), which is used as our baseline to benchmark the decoupled agents. The difference in performance caused small changes compared to the official ANAC 2011 ranking, as *Agent K2* moved up from 5th to 3rd place.

Table 2 in Appendix A provides an overview of the results. We evaluated the performance in terms of time of agreement and average overall utility. From these results, we can conclude that the variation is minimal: the largest difference between the original and decoupled agents is 0.010 for the average time of agreement (due to *Agent Smith*) and 0.009 for the average utility (due to *Hard-Headed*). Therefore the BOA agents and their original counterparts show comparable performance.

5. APPLICATIONS OF THE BOA FRAMEWORK

The BOA framework can be used to compare the performance of components and, using this knowledge, we can search for negotiation strategies that improve the current state-of-the-art. In this section we discuss a first exploratory test setup in which we change the acceptance condition and opponent model of existing ANAC 2011 agents to improve their performance.

Despite the reduced negotiation space that is searched, the space still needs to be scaled down. Decoupling n agents can in theory give rise to n^3 new agents if each agent implements all BOA components (see Figure 1), and even larger if we allow different parameters for each component. In practice, it quickly becomes unfeasible to search the full Cartesian product of components. To reduce the space, we have devised a method to test multiple acceptance criteria at the same time, as is explained below.

5.1 Scaling the Negotiation Space

Suppose that two negotiating BOA agents A and B have identical bidding mechanisms and the same opponent modeling technique, so that only their acceptance criteria differs. Furthermore, suppose agent A accepts in the middle of the negotiation, and agent B at the end. The agents have accepted at a different time during the negotiation, but the bidding behavior will be identical until the acceptance occurred. The only difference between the complete traces is that the trace of agent A is cut-off in the middle of the negotiation.

In the BOA framework we exploit this property by running all acceptance conditions in parallel, and recording when each acceptance condition accepts. This reduces the amount of component combinations from n^3 to n^2 as the n acceptance conditions are reduced to 1. This approach is from now on referred to as multi-acceptance criteria (MAC).

In addition, since we support parameters for the acceptance conditions, a large number of acceptance conditions varying only in the value of their parameters can be tested during the same negotiation thread. Note that this approach assumes that checking additional acceptance conditions does not introduce a large computational overhead. In practice we found that the computational overhead was less than 5%, even when more than 50 variants of acceptance conditions were used at the same time.

Note that a similar technique cannot be applied for the bidding strategy and the opponent model, as both components directly influence the negotiation trace.

Even if MAC is applied, there are still n^2 possible combinations to explore. This is already problematic for a limited amount of domains and agents. To illustrate, ANAC 2011 consists of 448 negotiation sessions [2] which may all last 3 minutes. In worst case, it requires 22 hours to run a single tournament, and almost four weeks for running it 28 times, as we did for the similarity test discussed in Section 4.2.2. Towards improving scalability, we extended GENIUS so that a negotiation tournament may be distributed among multiple computers.

5.2 Improving the State-of-the-Art

Using the scaling methods discussed in the previous section, we were able to explore a reduced space of negotiation strategies. We opted to limit our attention to the ANAC 2011 agents for two reasons: first, because it is a competition that already has verified data which can be re-used; second, the ANAC 2011 tournament is the most recent incarnation of ANAC at the time of writing, and can therefore be assumed to contain state-of-the-art negotiation agents.

5.2.1 *Searching the Negotiation Space*

For each agent of our test setup, the original bidding strategy was fitted with alternative acceptance conditions and opponent models. We used the following sets of BOA components:

(B) For the bidding strategies, the seven decoupled agents from ANAC 2011 were used (see Table 1).

(O) As our opponent model set we elected the two representative opponent models we identified in Section 4.1 (i.e., a Bayesian model and a frequency model). In addition, we allowed the strategies to use *no* opponent model, as the computational overhead of an opponent model could lead to worse performance.

(A) All acceptance conditions of the seven decoupled agents of the ANAC 2011 were used, except for the acceptance condition of *Gahboninho*, as it is relatively heavy-weight, resulting in poor cpu performance.

In addition to the existing acceptance mechanism components, we used acceptance mechanisms that combined certain acceptance criteria, such as $\mathbf{AC}_{\text{combi}}(T, \text{MAX}^W)$ [3], and the discounted version of $\mathbf{AC}_{\text{next}}$, called $\mathbf{AC}^{\text{disc}}_{\text{next}}(\alpha,\beta,\gamma,\delta)$. Similar to the acceptance condition of *IAMcrazyHaggler* [23], it differentiates between domains with and without discount factors; on undiscounted domains, it behaves identically to $\mathbf{AC}_{\text{next}}(\alpha,\beta)$ [3]; on the discounted domains it is equal to $\mathbf{AC}_{\text{next}}(\gamma,\delta)$. Table 4 in Appendix C provides an overview of all 83 tested acceptance conditions.

All possible combinations were run three times during an exploratory search to determine the best combination of components for each agent. Similar to the equivalence test, we replaced the original agent strategy by the new strategy and measured its performance in the ANAC 2011 tournament. The best agent strategies were run 10 times to determine whether the average utility was significantly improved.

5.2.2 Results

From the seven agents analyzed in the test set we were able to considerably improve four: *Gahboninho*, *Agent K2*, *Nice Tit for Tat*, and *BRAM Agent*. All four perform significantly better than their original (*two-tailed t-test*, $p < 0.01$).

For the other three agents, all the combinations of acceptance conditions and opponent models resulted in similar or worse performance. This indicates that the components of these strategies are well geared to one another. Note that this does not mean that the match is optimal, it only indicates that the components of the strategy are optimal within the tested set of components. Table 5 in Appendix D provides an overview of the performance of the best combination of components for each agent.

We note that using an improved opponent model does not ensure a better negotiation outcome, and in some cases can even result in worse performance due to the overhead caused by updating the model. An interesting direction for future work could be to quantify the contribution of opponent models to the performance of the ANAC agents.

All in all, the results demonstrate that the BOA framework not only assists in exploring the negotiation strategy space and improving existing agents, but it also helps to identify which components of the agent are decisive in its performance.

6. CONCLUSION AND FUTURE WORK

This paper introduces a framework that distinguishes the bidding strategy, the opponent model, and the acceptance strategy in automated negotiation strategies and recombines these components to systematically explore the space of automated negotiation strategies. The main idea behind the BOA framework is that we can identify several components in a negotiating agent, all of which can be optimized individually. Our motivation in the end is to create a proficient negotiating agent by combining the best components.

We have shown that many of the existing negotiation strategies can be re-fitted into our framework. We identified and classified the key components in them, and we have demonstrated that the original agents and their decoupled versions have identical behavior and similar performance. Finally, we have given an application of the BOA framework by recombining different components of the ANAC agents, and we have demonstrated this can significantly improve their performance.

One obvious direction of future research is to look at different BOA components in isolation; for example, to find the best opponent model that is currently available. After identifying the best performing components, we can turn our attention to answer whether combining effective components leads to better overall results, and whether an optimally performing agent can be created by taking the best of every component. Our framework allows us to make these questions precise and provides a tool for answering these questions.

Another possible improvement is extend the focus of current work on preference profile modeling techniques to a larger class of opponent modeling techniques, such as strategy prediction. Also, an agent is currently equipped with a single component during the entire negotiation session. It would be interesting to run multiple BOA components in parallel, and use recommendation systems to elect the best component at any given time.

Acknowledgements

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 VICI-project 08075.

7. REFERENCES

[1] R. Ashri, I. Rahwan, and M. Luck. Architectures for negotiating agents. In *Proceedings of the 3rd Central and Eastern European conference on Multi-agent systems*, pages 136–146. Springer-Verlag, 2003.

[2] Tim Baarslag, Katsuhide Fujita, Enrico H. Gerding, Koen Hindriks, Takayuki Ito, Nicholas R. Jennings, Catholijn Jonker, Sarit Kraus, Raz Lin, Valentin Robu, and Colin R. Williams. Evaluating practical negotiating agents: Results and analysis of the 2011 international competition. *Artificial Intelligence Journal*, Submitted.

[3] Tim Baarslag, Koen Hindriks, and Catholijn Jonker. Acceptance conditions in automated negotiation. In *Proceedings of The Forth International Workshop on Agent-based Complex Automated Negotiations (ACAN 2011)*, 2011.

[4] Tim Baarslag, Koen Hindriks, Catholijn M. Jonker, Sarit Kraus, and Raz Lin. The first automated negotiating agents competition (ANAC 2010). In Takayuki Ito, Minjie Zhang, Valentin Robu, Shaheen Fatima, and Tokuro Matsuo, editors, *New Trends in Agent-based Complex Automated Negotiations, Series of Studies in Computational Intelligence*, pages 113–135, Berlin, Heidelberg, 2012. Springer-Verlag.

[5] C. Bartolini, C. Preist, and N.R. Jennings. A generic software framework for automated negotiation. In *First International Conference on Autonomous Agent and Multi-Agent Systems*. Citeseer, 2002.

[6] C. Beam and A. Segev. Automated negotiations: A survey of the state of the art. *Wirtschaftsinformatik*, 39(3):263–268, 1997.

[7] R. Carbonneau, G.E. Kersten, and R.klaue2001automated Vahidov. Predicting opponent's moves in electronic negotiations using neural networks. *Expert Systems with Applications*, 34(2):1266–1273, 2008.

[8] M. Dumas, G. Governatori, A.H.M. Ter Hofstede, and P. Oaks. A formal approach to negotiating agents development. *Electronic Commerce Research and Applications*, 1(2):193–207, 2002.

[9] T. Eymann. Co-evolution of bargaining strategies in a decentralized multi-agent system. In *AAAI fall 2001*

symposium on negotiation methods for autonomous cooperative systems, pages 126–134, 2001.

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

[11] S.S. Fatima, M. Wooldridge, and N.R. Jennings. Multi-issue negotiation under time constraints. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pages 143–150. ACM, 2002.

[12] D.K. Gode and S. Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of political economy*, pages 119–137, 1993.

[13] K. Hindriks, C. Jonker, and D. Tykhonov. Towards an open negotiation architecture for heterogeneous agents. *Cooperative Information Agents XII*, pages 264–279, 2008.

[14] K. Hindriks and D. Tykhonov. Opponent modelling in automated multi-issue negotiation using bayesian learning. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, volume 1, pages 331–338. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[15] K.V. Hindriks and D. Tykhonov. Towards a quality assessment method for learning preference profiles in negotiation. *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*, pages 46–59, 2010.

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

[17] Shogo Kawaguchi, Katsuhide Fujita, and Takayuki Ito. Compromising strategy based on estimated maximum utility for automated negotiation agents competition (ANAC-10). In Kishan G. Mehrotra, Chilukuri K. Mohan, Jae C. Oh, Pramod K. Varshney, and Moonis Ali, editors, *IEA/AIE (2)*, volume 6704 of *Lecture Notes in Computer Science*, pages 501–510. Springer, 2011.

[18] Raz Lin, Sarit Kraus, Tim Baarslag, Dmytro Tykhonov, Koen Hindriks, and Catholijn M. Jonker. Genius: An integrated environment for supporting the design of generic automated negotiators. *Computational Intelligence*, 2012.

[19] N. Matos, C. Sierra, and N.R. Jennings. Determining successful negotiation strategies: An evolutionary approach. In *Proceedings of the International Conference on Multi Agent Systems*, pages 182–189. IEEE, 1998.

[20] Y. Oshrat, R. Lin, and S. Kraus. Facing the challenge of human-agent negotiations via effective general opponent modeling. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 377–384. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

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

[22] Liviu Dan Serban, Gheorghe Cosmin Silaghi, and Cristian Marius Litan. Agent fsega - time constrained reasoning model for bilateral multi-issue negotiations. In Takayuki Ito, Minjie Zhang, Valentin Robu, Shaheen Fatima, and Tokuro Matsuo, editors, *New Trends in Agent-based Complex Automated Negotiations, Series of Studies in Computational Intelligence*, pages 159–165, Berlin, Heidelberg, 2012. Springer-Verlag.

[23] C.R. Williams, V. Robu, E.H. Gerding, and N.R. Jennings. Iamhaggler: A negotiation agent for complex environments. 2010.

[24] D. Zeng and K. Sycara. Bayesian learning in negotiation. *International Journal of Human-Computers Studies*, 48(1):125–141, 1998.

APPENDIX

A. SIMILARITY TEST

	Avg. time of agr.	SD time of agr.	Avg. utility	SD utility
Agent K (Org.)	0.713	0.0057	0.666	0.0035
Agent K (Dec.)	0.714	0.0061	0.672	0.0045
Agent Smith (Org.)	0.469	0.0083	0.703	0.0041
Agent Smith (Dec.)	0.479	0.0053	0.707	0.0041
FSEGA (Org.)	0.425	0.0013	0.721	0.0009
FSEGA (Dec.)	0.426	0.0041	0.721	0.0024
IAMcrazyHaggler (Org.)	0.591	0.0103	0.699	0.0078
IAMcrazyHaggler (Dec.)	0.587	0.0069	0.702	0.0099
IAMhaggler2010 (Org.)	0.633	0.0110	0.682	0.0093
IAMhaggler2010 (Dec.)	0.636	0.0101	0.684	0.0066
Nozomi (Org.)	0.663	0.0071	0.704	0.0063
Nozomi (Dec.)	0.666	0.0062	0.708	0.0053
Yushu (Org.)	0.798	0.0030	0.715	0.0035
Yushu (Dec.)	0.800	0.0026	0.717	0.0037
Agent K2 (Org.)	0.619	0.0046	0.685	0.0040
Agent K2 (Dec.)	0.621	0.0050	0.686	0.0034
BRAM Agent (Org.)	0.683	0.0089	0.683	0.0054
BRAM Agent (Dec.)	0.687	0.0060	0.683	0.0033
Gahboninho (Org.)	0.667	0.0055	0.736	0.0044
Gahboninho (Dec.)	0.668	0.0053	0.742	0.0015
HardHeaded (Org.)	0.738	0.0009	0.758	0.0024
HardHeaded (Dec.)	0.735	0.0028	0.749	0.0034
IAMhaggler2011 (Org.)	0.494	0.0102	0.685	0.0023
IAMhaggler2011 (Dec.)	0.493	0.0078	0.683	0.0024
Nice Tit for Tat (Org.)	0.677	0.0078	0.676	0.0043
Nice Tit for Tat (Dec.)	0.683	0.0070	0.668	0.0025
The Negotiator (Org.)	0.716	0.0016	0.679	0.0027
The Negotiator (Dec.)	0.716	0.0014	0.679	0.0023

Table 2: The table shows performance (with standard deviation) of agents in an ANAC 2011 tournament before and after being decoupled.

B. RESULTS OF ANAC COMPETITION

Agent	Amsterdam Trip	Camera	Car	Energy	Grocery	Company Acquisition	Laptop	Nice or Die	Mean utility
HardHeaded	0.891	**0.818**	**0.961**	0.664	0.725	0.747	0.683	**0.571**	0.757
Gahboninho	**0.912**	0.659	0.928	**0.681**	0.667	0.744	0.726	**0.571**	0.736
Agent K2	0.759	0.719	0.922	0.467	0.705	0.777	0.703	0.429	0.685
IAMhaggler 2011	0.769	0.724	0.873	0.522	0.725	**0.814**	**0.749**	0.300	0.685
BRAM Agent	0.793	0.737	0.815	0.420	0.724	0.744	0.661	**0.571**	0.683
The Negotiator	0.792	0.744	0.913	0.524	0.716	0.748	0.674	0.320	0.679
Nice Tit for Tat	0.733	0.765	0.796	0.508	0.759	0.767	0.660	0.420	0.676
Value Model Agent	0.839	0.778	0.935	0.012	**0.767**	0.762	0.661	0.137	0.611

Table 3: ANAC 2011 results of our hardware (n = 10).

C. VARIABLES USED FOR ACCEPTANCE CONDITIONS

Acceptance Condition	**Range**	**Increments**
$\mathbf{AC}_{\text{maxinwindow}}(T)$	T$\in$ [0.95, 0.99]	0.01
$\mathbf{AC}^{\text{disc}}_{\text{next}}(\alpha,\beta,\gamma,\delta)$	$\alpha \in$ [1.0, 1.05]	0.05
	$\beta \in$ [0.0, 0.1]	0.05
	$\gamma \in$ [1.0, 1.1]	0.05
	$\delta \in$ [0.0, 0.15]	0.05
$\mathbf{AC}_{\text{HardHeaded}}$	–	–
$\mathbf{AC}_{\text{TheNegotiator}}$	–	–
$\mathbf{AC}_{\text{NiceTitForTat}}$	–	–
$\mathbf{AC}_{\text{BRAMAgent}}$	–	–
$\mathbf{AC}_{\text{AgentK2}}$	–	–
$\mathbf{AC}_{\text{IAMhaggler2011}}$	–	–

Table 4: Variables that were used for the acceptance conditions.

D. IMPROVED AGENTS STRATEGY RESULTS

Agent	Original Utility	Best Performing Opponent Model	Best Performing Acceptance Condition	Improved Utility
Gahboninho	0.736	Original Model (No Model)	$\mathbf{AC}^{\text{disc}}_{\text{next}}(\alpha, \beta, \gamma, \delta)$ α: 1.0; β: 0.0; γ: 1.1; δ: 0.1;	0.759
Agent K2	0.685	IAMhaggler Model	$\mathbf{AC}^{\text{disc}}_{\text{next}}(\alpha, \beta, \gamma, \delta)$ α: 1.0; β: 0.0; γ: 1.0; δ: 0.15;	0.724
BRAM Agent	0.683	Original Model (No Model)	$\mathbf{AC}^{\text{disc}}_{\text{next}}(\alpha, \beta, \gamma, \delta)$ α: 1.0; β: 0.05; γ: 1.1; δ: 0.1;	0.697
Nice Tit For Tat	0.676	Original Model (Bayesian Model)	$\mathbf{AC}_{\text{maxinwindow}}(t)$ t: 0.99	0.696
HardHeaded	0.757	Original Model (Frequency Model)	$\mathbf{AC}_{\text{HardHeaded}}$	–
TheNegotiator	0.679	Original Model (No Model)	$\mathbf{AC}_{\text{TheNegotiator}}$	–
IAMhaggler2011	0.685	Original Model (No Model)	$\mathbf{AC}_{\text{IAMhaggler2011}}$	–

Table 5: Results of the improved agent strategies in an ANAC 2011 tournament (for n = 10 runs). The first four agents were improved significantly. Ther other two agents did not improve significantly in our test setup.