Negotiation Agent Final Report, Group 18

Wesley Quispel
Technical University Delft
Delft, Netherlands
W.Quispel@student.tudelft.nl

Joris Verhagen
Technical University Delft
Delft, Netherlands
J.P.M.Verhagen@student.tudelft.nl

Stavya Bhatia
Technical University Delft
Delft, Netherlands
S.Bhatia-2@student.tudelft.nl

Anna Mészáros
Technical University Delft
Delft, Netherlands
A.Meszaros@student.tudelft.nl

Cristian Meo
Technical University Delft
Delft, Netherlands
C.Meo@student.tudelft.nl

1 INTRODUCTION

With new technological advancements in the field of smart assistants, more focus is being directed towards fields where people are considered to excel at. One such field is communication, and even more specifically, negotiations. While people are great communicators, their decisions are not always rational, which in turn can make them pass up good offers. To aid in this, negotiation agents are being developed to represent their user's preference on a given subject. Over the course of this project, an agent has been developed that aims to obtain the best deal for itself. In order to achieve this, strategies for making and accepting bids were developed to respond according to the opponent's behaviour. Since the preferences of the opponent are unknown, it was also imperative to make an approximation based off of the information that the opponent gave through their bids. Great difficulties arise when negotiation agents represent users on domains where the users do not have perfect numerical preference indicators. To ensure that the agent is capable of negotiating under preference uncertainty of the user an additional user preference elicitation technique is applied. The full scope and capabilities of the agents are described within this paper.

2 NEGOTIATION STRATEGY

2.1 Strategy Components

A negotiation strategy generally consists of four components - the acceptance strategy, the bidding strategy, the opponent model and the opponent model strategy. These four components have been defined as follows [2]:

Acceptance Strategy: determines if an opponent's bid is acceptable or not. There are two common approaches for the implementation of the acceptance strategy. One is the use of a simple utility threshold. This may be varied by taking the remaining time into account. The second approach is the use of a linear function which takes the offer that the agent is planning to offer into account. If the opponent's offer has a higher utility than the next offer being considered by the agent, the opponent's offer is seen as acceptable.

Bidding Strategy: determines which bid will be placed next by the agent. Common strategies include time-dependent, behaviour-dependent and zero-intelligence strategies. An example of a time-dependent strategy is the Boulware tactic, where the agent only begins conceding when the deadline starts approaching. Tit for Tat [1] is an example of a behaviour-dependent strategy. It is a reciprocal strategy in which the agent mirrors the opponent's behaviour. For example, when the opponent concedes, so does the agent, if the opponent plays though, the agent responds alike. Lastly, the SimpleAgent analysed in 1.1.1 is an example of a zero-intelligence strategy as it does not take the opponent's actions nor any other factors into account when making a bid.

Opponent Model: is an estimate of the opponent's true behaviour based on the behaviour observed by the agent. Opponent models aim to estimate the opponent's utility for future bids, but can also be expanded to estimate the opponent's preference profile [5], its strategy and in turn the opponent's future moves [3]. There are two main types of opponent models - frequency-based models and Bayesian models. The frequency model estimates the opponent's preferences based off of the number of times issues and values occur, whereas the Bayesian model is updated through Bayesian learning.

Opponent Model Strategy: determines how the opponent model is used to influence the bidding strategy. This of course, in part, depends on the type of bidding strategy being implemented. For example, if a conceder-like strategy is being used, the opponent model could be used to estimate the Pareto optimal frontier and stay as close to it as possible so that both parties reach a mutually beneficial deal.

2.2 The Acceptance Strategy

The acceptance strategy of the "negotiator" follows a twofold methodology based on whether the opponent agent is assumed to be conceding or not. First, a Boolean concession variable is evaluated. This is done by comparing the latest opponent bid over a set horizon of opponent bids. If the latest bid has not occurred in the horizon, the opponent is identified as conceding and vice versa. Further, if the concession variable is found to be True/False then the opponent is assumed to be conceding/not conceding.

Not Conceding Phase: In the case that the opponent is not conceding, the agent follows a hybrid time dependent approach to accept or reject the offer. Till such time the duration of the negotiation is less than 84% of the total duration, the agent accepts the offer above a utility threshold that is tantamount to the mean of the top 15 percentile of the total possible bids based on utility. However, for time greater than 84% of the total duration the agent uses a discount factor to discount the threshold utility for acceptance. The discount factor is given as:

$$Utility.Threshold.Discount = Utility.Threshold * (1.832 - t) \quad (1)$$

where $Utility.Threshold$ is the mean of the top 15 percentile of the total bids based on utility and t is the time elapsed.

Conceding Phase: In the second case, where the opponent is identified as conceding, the BOA agent will accept or reject the opponent bid based on a function value. The function value is calculated as,

$$ValueFunction = \frac{U_a(k+1)^2}{U_a(k-1)} - U_o(k)\frac{N + log(t \cdot \sqrt{2})}{N} \quad (2)$$

where the $U_a(k+1)$ is the utility of the agent's next bid, the $U_a(k-1)$ is the utility of the agent's previous bid, the $U_o(k)$ is the utility of the opponent's current bid, and N and t represent the total time and current time of negotiation respectively.

This function value attempts to compare the agent's next bid with the opponent's current bid after accounting for-
1) direction the agent is intending to take in the future
2) the time left for the negotiation;
into the bid utilities itself. This is done by augmenting the next bid of the agent by an index value that is the ratio of the agent's next bid to the agent's previous bid. In addition, it also augments the opponent's bid with an index factor that increases with time as shown in the formula above. Based on the value returned by the function the agent either accepts the opponent's bid (value function<0) or continues the negotiation (value function>=0) for a better prospect in the future.

2.3 The Bidding Strategy

The bidding strategy being proposed is that of a deceitful agent, which attempts to hide its true preferences from the opponent and maximise its own utility. It consists of three phases: *the learning phase*, *the leverage phase*, and *the last resort*, each of which will be explained in further detail. An overview is shown in Figure 1 where the phases are numbered with 1, 2, and 3 respectively. The bounds and bids in the figure merely serve an indicative purpose and are dramatized for clarity.

The Learning Phase: Since nothing is known of the opponent at the start of the negotiation, the goal of this phase is to learn an accurate model of the opponent's preferences, in order to act on those during the later part of the negotiation. The opponent model can later be utilised in order to determine if the opponent is conceding, or determine their reservation value and use this information in order to get a better deal. With this consideration in mind, the learning phase is also used in order to prevent the opponent from accurately estimating the agent's profile. This is achieved by performing bounded Random Walking.

Two thresholds are established at the beginning of the negotiation by manually deciding on a range around the upper- and lower bound and determining a utility bound within that range. The thresholds with their respective ranges allow for enough bids in the *leverage phase*, the *learning phase* and ensuring that a bid

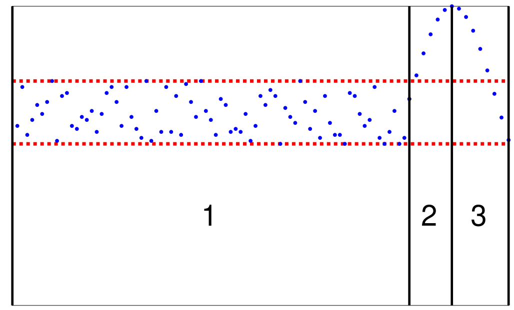

Figure 1: A qualitative general outline of the bidding strategy indicating the three bidding phases. The red dotted lines denote the upper and lower utility bounds within which bids in the lying phase are generated

made during the *learning phase* is always of a permissible utility if it were to be accepted before the other phases take place. As an example, the lower threshold's lower bound is the lowest utility that the agent is willing to get accepted. The lower threshold's higher bound is chosen to be significantly above that. The actual threshold is determined by checking whether 40 percent of the bids are above the lower bound and below the upper bound (the agent would use the top 10 percent of the bids for the *leverage phase* and 30 percent of the bids for the *learning phase*). If this is the case, the actual utility of 40 percent of the bids gets chosen, otherwise one of the bounds is chosen.

The upper and lower bound, shown as the two red dotted lines in Figure 1, give the utility range that the offers can have, while also being within an acceptable utility. Over the course of the learning phase, a random utility within these bounds is chosen and a bid with a utility closest to this random utility is chosen. Random Walking carries a risk of offering the opponent a bid with a high utility for them. In order to minimise this risk, the opponent model, although incomplete, is used to estimate whether the random bid has a high utility for the opponent (in our particular implementation, anything above 0.7 for is considered a high utility for the opponent during the learning phase). If this is the case, a new random bid is generated, thus simultaneously adding a third bound. The generation of a new random bid is repeated a fixed number of times in order to avoid endless loops, in case it is not possible to generate a bid under the estimated bid utility of the opponent. The learning phase ends once 85 percent of the negotiation time has passed, or it is detected that the opponent has started conceding.

Registering whether the opponent has started conceding is done by observing the opponent's last couple of bids, and seeing if the latest bid has appeared in the subset before. Conceding is characterised by an agent lowering their utility, which in turn means that a bid is unlikely to occur twice during a concession. After the learning phase, the leverage phase begins.

The Leverage Phase: is the phase in which the agent attempts to maximise its utility. Given that the opponent's model of the agent is determined on previously made bids, it has no way of distinguishing between an agent conceding and an agent increasing their utility. Furthermore, as time passes, most agents are more likely to begin conceding rather than increasing their utility. In turn an opponent is more likely to concede and accept a deal being made. Meanwhile, rather than conceding, the agent is increasing its utility step by step, thus feigning concession while moving towards its optimal bid.

In order to maintain this illusion of concession for as long as possible, the agent raises its utility in steps in accordance to the following formula:

$$U(k) = \frac{bidsInRange}{bidsAvailable} speedFactor + U(k-1) \tag{3}$$

Here, *bidsInRange* denote the number of bids available in the utility space ranging from the previous bid utility to the maximum bid utility. *bidsAvailable* indicate the number of bids the agent is physically capable of making in the time remaining. This is estimated based off of the number of bids made so far in the elapsed time. The *speedFactor* is determined based off of the opponent's previous bids in the following manner:

$$speedFactor = \begin{cases} \dfrac{U_o(k-2) + U_o(k-1)}{|log(|U_o(k-2) + U_o(k-1)|)|}, & \text{if } k > 2 \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

where U_o is the opponent's utility for a given time step. The aim of this is to adapt the agent's lying speed to the opponent's concession speed. This reduces the chance of opponents with high concession speeds accepting a bid before the agent reaches its optimal utility. Furthermore, in the case of opponents with low concession speeds it also reduces the chance of quickly reaching the optimal bid and remaining there for a long period of time, thus potentially breaking the illusion that the agent is conceding.

Nonetheless, certain opponents are more "hard-headed" than others, which can lead to time running out before they decide to accept a deal below their preferred utility. For this reason, the last resort was introduced as a back-up plan.

The Last Resort: takes effect once 96% of the negotiation time has passed, and the opponent has made 5 or more bids. The latter is only relevant for very short negotiations in which less than 5 bids are possible, yet nonetheless important given that the opponent's average utility is approximated with the help of the last five bids in this particular implementation of the strategy. At this point the agent begins conceding based off of the opponent's concession behavior. If the opponent is thought to be conceding quickly, our agent concedes quickly too. The utility of the new bid is determined by Equation (5)

$$U(k) = U(k-1) - \frac{|U_{o,avg} - U_o(k-1)|}{e^{\alpha} + 1} \tag{5}$$

$$\alpha = \begin{cases} |100 - 100T| \cdot \dfrac{|U_{o,avg} - U_o(k-1)|}{U(k-1)}, & \text{if } T \leq 0.99 \\ |100 - 100 \cdot 0.99| \cdot \dfrac{|U_{o,avg} - U_o(k-1)|}{U(k-1)}, & \text{otherwise} \end{cases} \tag{6}$$

Where $U(k)$ indicates the utility of the new bid, $U(k-1)$ and $U_o(k-1)$ indicate the utility of the last bid for the agent and opponent respectively, $U_{o,avg}$ is the opponents average utility, and T is time of the negotiation session.

While conceding, the Opponent Model Strategy is implemented in order to find the bid with the highest utility for the opponent in the range $[U(k-1), U(k)]$, whereby $U(k-1) > U(k)$. At the same time, the aim is to receive a better final utility than the opponent, which is why it is also checked that $U(k) > U_o(k)$ is true. If this is not the case, the range from which the opponent's highest utility bid is selected from is narrowed to $[U(k-1), U'(k)]$, whereby $U(k-1) > U'(k)$ and $U'(k) > U(k)$. Nonetheless, in both cases, this approach increases the chance of reaching a Pareto Optimal bid, in other words, a bid from which there exists no other bid where both parties profit.

3 OPPONENT MODEL

The Opponent Model aims to give insight in which issues are important for the opponent, what its preferences are (by means of estimating the issue values), and learning about the opponent's strategy. A sufficient Opponent Model allows the agent to make bids that are with a higher utility for the opponent while remaining with an equal utility for the agent itself (essentially moving along the Pareto optimal line). It is then more likely that the opponent accepts the offer. It also allows the agent to evaluate the offer it wants to make with respect to the opponent's estimated utility profile to ensure that an offer never has a higher utility for the opponent than it has for the agent (especially useful in the tournament setting of the ANAC competition).

The implemented Opponent Model is based on Frequency Analysis Heuristics which is based on how often values of a particular issue change and the frequency of appearance of values in offers. When values inside a particular issue change often it is an indication that the issue has a low weight for the opponent. If a particular value appears often in the opponent's bids it is an indication that the value has a high utility.

3.1 Values

The utility of the values for a certain issue are based on the frequency of occurrence in the opponent's bid. Every received bid from the opponent is analyzed on its content and a count is being updated that keeps track of the number of occurrences for a certain value. The utility of a value is based on the assumption that the agent's own preference profile, and therefore the utilities for values of a specific issue, are comparable to the opponent's preference profile. This assumption is supported by analysis of different preference profiles for different domains. The range of and distribution for the issue's values are comparable for different preference profiles (only the values, the values coupled to specific issues are of course not).

The count that was being kept track off is used in this mapping of count to issue's values. The value with the highest occurrence (and therefore count) in the opponent's bid history is given the agent's highest value for that issue as an estimate of the opponents value for that issue. The value with the lowest count is given the agent's lowest value for that issue. The algorithm then moves to the value with the second highest count and gives it the agent's second highest utility for that issue. This pattern is continued until all issues have been dealt with. If multiple issues share a certain count they are given the same value. Based on the back-and-forth nature of the algorithm, it might happen that values in the middle of the agent's preference profile are not used in the opponent's preference profile assessment. It was determined that not implementing these values is of less importance than not implementing either the highest or the lowest values. The algorithm shown below shows the process in pseudo-code which assumes that a hashmap with the counts and a hashmap with the agent's own utility values per issue are already obtained. The process of obtaining these maps is rather straightforward so only the mapping from count hashmap to opponent utility estimates is shown here.

Algorithm 1: Utility Value Mapping

```
Procedure: Map Utilities
Input: uPerIssue hashmap with all of the agent's utility
  values per issue. vCntMap hashmap with count of value
  occurrences.
init: oppU hashmap containing mapping of agent's utilities
  per issue to valueCountMap.
uPI_c ← uPerIssue;
vCM_c ← vCntMap;
for all issues do
    for all values in issue do
        maxU ← max(vCM_c.issue);
        all oppU.issue.value(maxU) ← max(uPI_c.issue)
        max(uPI_c.issue).pop();
        max(vCM_c.issue).pop();
        minU ← min(vCM_c.issue);
        all oppU.issue.value(minU) ← min(uPI_c.issue)
        min(uPI_c.issue).pop();
        min(vCM_c.issue).pop();
    end
end
return: oppU
```

3.2 Weights

When opponent bids do not change from bid to bid it is assumed that that specific value is more important for the opponent profile. This lays at the foundation of modeling the opponent's utility using a frequency heuristic. As such, whenever a value does not change for a particular issue from bid to bid we add a constant value $\epsilon = 0.2$ to the already estimated issue weight. After normalizing the aforementioned issue weight to obtain a list of weight that sum up to one, we can observe that issues that change relatively little have the highest weight associated with them.

3.3 Utility Estimation

We now have a rough estimation of the issue values by having counted them, in addition to the rough weight estimation. Estimating the opponent's utility for a bid (whether this is from the agent or from the opponent does not matter) is done by multiplying the value occurring in the bid with the normalized weight of the corresponding issue and summing over all issues in the bid according to Formula. 7.

$$U(X_1, X_2, ..., X_n) = \sum_{i=1}^{n} w_i \cdot U_i(X_i) \tag{7}$$

This allows for both estimating the utility of the opponent's bid as well as estimating the utility of the bid that the agent itself is about to make.

4 OPPONENT MODEL STRATEGY

The Opponent Model Strategy aims to utilise the knowledge gained from the Opponent in order to better plan the agent's own bids. The chosen Opponent Model Strategy returns the best bid of a set of bids from the opponent's point of view. The information is then utilised in the bidding strategy in order to improve the concession part of the strategy. This is done with the intention of finding the best bid for the opponent within a utility range around the agent's next planned bid, so as to find a bid that the opponent is more likely to accept and thus reach an agreement as soon as possible.

Another aspect of the Opponent Model Strategy is the flag for whether the Opponent Model can be updated. In this particular case, the flag returns true as long the negotiation is in the first half. This approach was chosen so as to give the Opponent Model enough time to gather information on the opponent, whilst aiming to prevent the quality of the model dropping once the agents begin conceding as this is the point when their true preferences are no longer being represented.

5 HANDLING PREFERENCE UNCERTAINTY

There are circumstances in which the exact preferences of the user are unknown to the agent itself. The only information is a ranking of bids, based on their relative utility. In order to deal with such a situation, a form of frequency model has been implemented for estimating the values and weights of each attribute.

5.1 Attribute Preference Values

The steps for eliciting the values are as follows:

(1) Determine the number n of bids within the ranking.
(2) Create an importance estimate factor. This places more importance on values that occur higher in the list, since these bids have a higher utility. The bid ranking is sorted in ascending order, and as such the final bid has an importance factor of n, and prior bids have an importance equal to $n - i$ where i the ith bid from the most important bid.
(3) Count the occurrences of a value, while weighing them with the importance estimate factor. In other words, preference

values across all attributes in the most important bid contribute a value of n, the values in the previous bid contribute $n-1$, and so forth. This incorporates both the importance of the bid as in a frequency model and the preference as given across bid ranking.
(4) Group the value counts with their value names under the same issue (e.g. all the food together, all the drinks together, etc.).
(5) Normalise by dividing with the maximum value within each group. The normalised values are the estimated issue values.

5.2 Attribute Weights

In order to elicit the weights, the following algorithm was implemented:

Algorithm 2: Weight Elicitation

```
Procedure: Estimate weights
Input: ranking list of bids sorted in ascending order
  according to relative utility.
init: weightMap hashmap containing mapping of agent's
  issues to the estimated weights.
unchanged ← false
counter list of number of times a value has not changed
  within an issue
for all bids do
    for all issue in bid do
        if issue(bid)==issue(bid-1) then
            bid.unchanged = true
        end
        issue.counter = count(unchanged == true)
    end
end
weightMap.weights = normalise(counter) return:
  weightMap
```

5.3 Performance

Observing the movement of the agent of the agent through the utility space for varying degrees of uncertainty, it can be observed that the agent displays behaviour closer to the expected behaviour as the preference uncertainty drops. This is to be expected since the amount of data dictates how accurate and estimation will be. The main issue within the preference uncertainty setting is that the agent has a tendency to overestimate the utility of bids. This leads the agent to offer bids below what it would actually offer given it knows its preferences. Nonetheless, the agent is still capable of winning against standard agent types such as Boulware and Conceder.

6 PERFORMANCE ANALYSIS

6.1 Efficiency

One of the most important aspects of a trusted negotiation agent is that the agent performs well and as expected for different utility profiles. This property is often seen as the efficiency of the agent. To assess the efficiency of the BabyGenius negotiation agent, the performance was tested against itself, the Boulware Agent, the Conceder Agent,the Hardliner agent and the Ponpoko Agent (winner of ANAC 2017), using three different utility profiles. In every simulation the first utility profile is referred to the agent and the second utility profile to the opponent. All simulations have been done on the Party Domain, which contains 6 issues with each having 3 to 4 values.

BabyGenius versus BabyGenius First the BabyGenius negotiation agent was tested against itself for the three different profiles. From Table. 2 it can be seen that an agreement is never reached. This is due to the Tit-for-Tat behaviour of the agent which make it not concede if the opponent is not showing any willingness to do it as well. In this environment it is unwanted to obtain an agreement with low utility for the agent and high utility for the opponent since this helps the opponent win the tournament.

utility profile	u1	u2	u3
Time	3,000	3,000	3,000
Rounds	2950	2961	2868
Agreement	No	No	No
Dis to Pareto	1,035	1,085	1,236
Dis to Nash	1,206	1,210	1,337
Social Welfare	0	0	0
Agent utility	0	0	0
Opp utility	0	0	0

Table 1: BabyGenius versus BabyGenius for three different utility profiles

BabyGenius versus Boulware
Secondly the BabyGenius negotiation agent was tested against the Boulware Agent for the three different utility profiles. Notice in Table. 2 that here in every negotiation an agreement was reached. Every agreement was in the BabyGenius' favour. On Fig. 2 it can be seen that in the first two cases there are no relevant differences, while in the third case the opponent's utility is almost as high as the agent's utility. Indeed, in Table. 2 it can be noted that in this case the distance from the Nash-equilibrium point is small. Moreover, in this case the agreement has been reached on the Pareto frontier, therefore an efficient outcome has been reached.

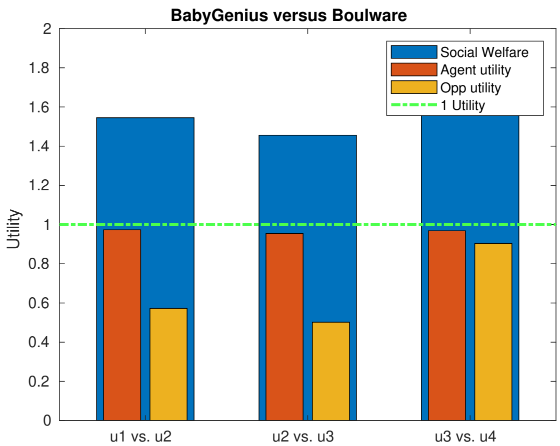

Figure 2: BabyGenius versus Boulware Agent on three different domains. Victory for all three utility profiles.

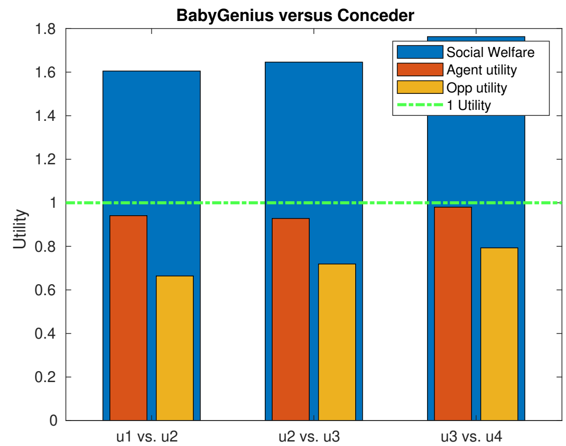

Figure 3: BabyGenius versus Conceder for three different utility profiles

utility profile	u1	u2	u3
Time	2,495	2,055	1,946
Rounds	2223	2277	2197
Agreement	Yes	Yes	Yes
Dis to Pareto	0,014	0,024	0,000
Dis to Nash	0,277	0,366	0,034
Social Welfare	1,545	1,456	1,872
Agent utility	0,973	0,954	0,968
Opp utility	0,572	0,502	0,904

Table 2: BabyGenius versus Boulware for three different utility profiles

BabyGenius versus Conceder

Thirdly the BabyGenius negotiation agent was tested against the Conceder Agent for three different utility profiles. From Table 3 it can be seen that for all utility profiles the agreement has been reached. However, none of them are on the Pareto frontier. Therefore no efficient outcome has been reached.

utility profile	u1	u2	u3
Time	0,341	0,245	0,132
Rounds	331	206	134
Agreement	Yes	Yes	Yes
Dis to Pareto	0,020	0,012	0,032
Dis to Nash	0,246	0,154	0,187
Social Welfare	1,605	1,646	1,763
Agent utility	0,941	0,928	0,980
Opp utility	0,664	0,719	0,793

Table 3: BabyGenius versus Conceder

BabyGenius versus Hardliner

Then the BabyGenius negotiation agent was tested against the Hardliner agent. From Table .4 it can be seen that the agreement is never reached. This happens because the Hardliner agent does not concede, therefore since the BabyGenius agent displays the Tit-for-Tat behaviour and the ranking method in the tournament setting is taken into account, an agreement cannot be reached.

utility profile	u1	u2	u3
Time	3	3	3
Rounds	1665	2765	2709
Agreement	0	0	0
Dis to Pareto	1.03603	1.08556	1.23586
Dis to Nash	1.2158	1.20458	1.33368
Social Welfare	0	0	0
Agent utility	0	0	0
Opp utility	0	0	0

Table 4: BabyGenius versus Hardliner

BabyGenius versus Ponpoko

Lastly the BabyGenius negotiation agent was tested against the Ponpoko Agent, which won ANAC 2017, for the three different utility profiles. Table. 5 shows much closer results for the obtained results. A closer look at the table reveals that just in some cases an agreement is reached. Where it is not indicated it means that just in some simulations an agreement has been reached. This is the reason why on Fig. 4 the represented utility is so low. However, it can be seen that the agent's utility is on average higher than the opponent one. When an agreement is reached it is often on the Pareto Optimal frontier. Additionally, when the agreement is made, it is within high proximity of the Nash-Equilibrium point. Finally, when an agreement is made, it is often on the Pareto Optimal Frontier, therefore in some cases an efficient outcome is reached.

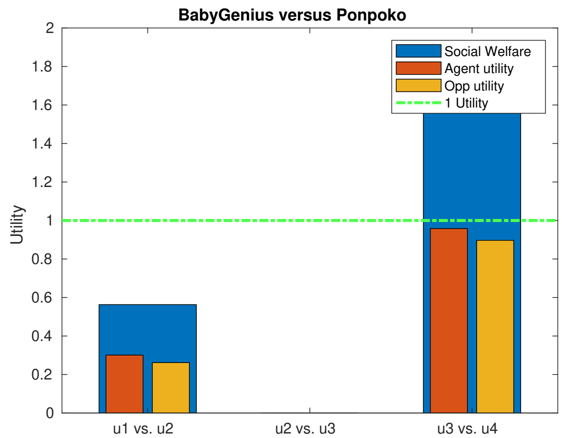

Figure 4: BabyGenius versus Ponpoko for three different utility profiles

utility profile	u1	u2	u3
Time	2,330	3,000	2,360
Rounds	3284	3297	2354
Agreement	Yes	No	Yes
Dis to Pareto	0,691	1,086	0,000
Dis to Nash	0,819	1,205	0,062
Social Welfare	0,563	0,000	1,855
Agent utility	0,301	0,000	0,958
Opp utility	0,262	0,000	0,897

Table 5: BabyGenius versus Ponpoko

6.2 Robustness

To create a trusted negotiation agent that can show parties that it is usable to conduct negotiations for them it is important that the agent performs well and as expected on different domains. This property is often seen as the robustness or adaptability of the agent. To assess the robustness of the BabyGenius negotiation agent, the performance was tested against itself, the Boulware Agent, the Conceder Agent, and the Ponpoko Agent (winner of ANAC 2017), on three different domains. These domains were chosen as the Party domain, the WindFarm domain, and the Jobs domain. All three domains where chosen due to their sufficient but different amount of possible bids and of the utility differences between bids. The Party Domain contains 6 issues with each 3 to 4 values, the WindFarm Domain contains 7 issues with each 2 to 4 values, and the Jobs Domain contains 6 issues with each 2 to 5 values. These domains therefore test the performance of the BabyGenius negotiation agent in different settings. On each domain the negotiation was run three times, each time with a random utility profile to obtain results representative for the entire domain and not just for the utility profile - domain combination. The tables below show the averaged results.

BabyGenius versus BabyGenius

First the BabyGenius negotiation agent was tested against itself on the three different domains. Notice from Table. 6 that an agreement is never reached. This is since the Tit-for-Tat properties of the agent make it not decrease its own utility if the opponent is not showing any willingness to lower theirs. Although unwanted when trying to find a solution in the real world, this behavior is modeled due to the tournament setting that the BabyGenius negotiation agent will participate in. In this environment it is unwanted to obtain an agreement with low utility for the agent and high utility for the opponent since this helps the opponent win the tournament.

Domain	Party	Windfarm	Jobs
Time	3,000	3,000	3,000
Rounds	1404	755	1842
Agreement	No	No	No
Dis to Pareto	1,036	1,093	1,009
Dis to Nash	1,216	1,226	1,042
Social Welfare	0,000	0,000	0,000
Agent utility	0,000	0,000	0,000
Opp utility	0,000	0,000	0,000

Table 6: BabyGenius versus BabyGenius

BabyGenius versus Boulware

Secondly the BabyGenius negotiation agent was tested against the Boulware Agent on the three different domains. Notice in Table. 7 that here in every negotiation an agreement was reached. Every agreement was in the BabyGenius' favour. Looking at the Job domain we see a lower utility value for BabyGenius. This is caused due to the steep utility differences for bids near 1.0 utility. This causes the agent to remain on the lower bid for longer since it waits until the perfect moment for the 1.0 utility bid to be offered. In this simulation, some of the results on the Party domain and the Jobs domain were on the Pareto Frontier. Notice also in Table 7 that the distance to the Pareto Frontier is low for all domains. None of the agreements were at the Nash-equilibrium. The distances of agreement to the Nash equilibrium are significantly higher than those for the Pareto Frontier.

Domain	Party	Windfarm	Jobs
Time	2,441	2,561	2,769
Rounds	3245	2489	3212
Agreement	Yes	Yes	Yes
Dis to Pareto	0,023	0,034	0,037
Dis to Nash	0,071	0,228	0,306
Social Welfare	1,664	1,589	1,327
Agent utility	0,947	0,970	0,886
Opp utility	0,718	0,620	0,441

Table 7: BabyGenius Agent versus Boulware Agent

The figure below shows the Agent's utility, the opponent's utility and the social welfare (added utility) averaged for the three simulations on each domain. This clearly shows that the lower utility in the Job domain also causes a lower utility for the opponent and thus for the social welfare. An undesirable characteristic in real-world negotiations but irrelevant for the tournament setting since a decisive victory is still achieved.

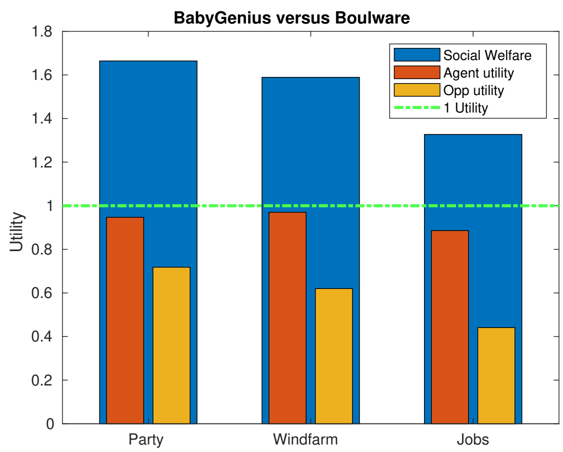

Figure 5: BabyGenius versus Boulware Agent on three different domains. Victory on all three domains

BabyGenius versus Conceder

Thirdly the BabyGenius Agent was tested against the Conceder Agent on the three different domains. The Conceder Agent, as the name suggests, starts conceding earlier than the Boulware agent allowing for a higher utility for the BabyGenius. In this simulation, some of the results on the Party domain and the Jobs domain were on the Pareto Frontier. None of the agreements were at the Nash-equilibrium. The BabyGenius agent is less able to achieve theoretical optimal bids on the Windfarm domain as the distances to either the Pareto Frontier or the Nash equilibrium suggests. This is likely due to the fact that the fast conceding makes it unpredictable when the opponent will accept the agent's bid which makes it less likely to make the lying phase (as shown in Figure. 1) work exactly as intended.

Domain	Party	Windfarm	Jobs
Time	0,577	1,880	1,474
Rounds	383	1358	1202
Agreement	Yes	Yes	Yes
Dis to Pareto	0,016	0,260	0,009
Dis to Nash	0,093	0,479	0,391
Social Welfare	1,669	1,369	1,344
Agent utility	0,968	1,000	0,913
Opp utility	0,701	0,369	0,430

Table 8: BabyGenius Agent versus Conceder Agent

The figure below shows the Agent's utility, the opponent's utility and the social welfare (added utility) averaged for the three simulations on each domain. Similarly to Figure. 5, Figure. 6 shows the lower utility in the Job domain for both the BabyGenius agent and the Opponent. Overall the performance is better than against the Boulware agent. Due to the different characteristics between the Boulware- and Conceder agents (late conceding versus early conceding) these results are expected.

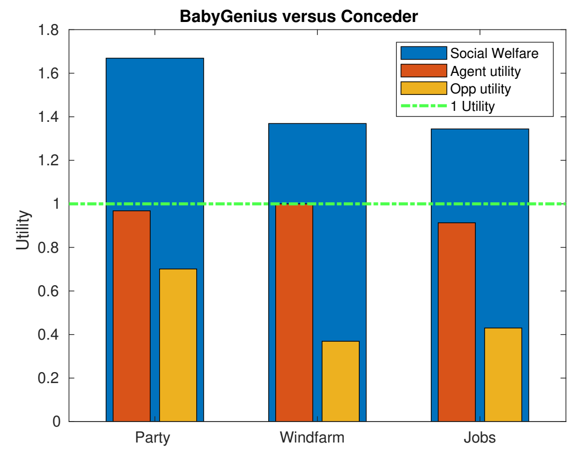

Figure 6: BabyGenius versus Conceder Agent on three different domains. Victory on all three domains

BabyGenius versus Ponpoko

Lastly the BabyGenius negotiation agent was tested against the Ponpoko Agent, which won ANAC 2017, on the three different domains. Due to the elementary strategies applied in the Boulware- and Conceder agent and their relative simple workings, it was determined to be of great importance to test the BabyGenius agent against an agent comparable to one that might be in the tournament. Table. 9 also shows much closer results for the obtained utility. When an agreement is reached it is always on the Pareto Optimal frontier and for 2 out of 3 negotiation sessions on the Party Domain it was at the Nash equilibrium. This shows that the BabyGenius agent performs not necessarily better but more theoretically optimal against complex agents.

Domain	Party	Windfarm	Jobs
Time	2,562	2,996	3,000
Rounds	2948	1400	4043
Agreement	Yes	Sometimes	No
Dis to Pareto	0,000	0,758	1,009
Dis to Nash	0,013	0,861	1,042
Social Welfare	1,805	0,590	0,000
Agent utility	0,912	0,300	0,000
Opp utility	0,893	0,290	0,000

Table 9: BabyGenius Agent versus Ponpoko Agent

Figure. 7 shows the plotted results. This figure also shows that on the Party Domain all three simulations resulted in an agreement. On the WindFarm domain only one simulation resulted in an agreement and on the Jobs domain none of the simulations resulted in an agreement. Out of the 4 simulations that resulted in an agreement the BabyGenius agent has won 3. In the aforementioned tournament setting this is a good result since it shows that the BabyGenius agent does not give too much utility away for free (sometimes it does not even allow an agreement to be made) and that it mostly wins if it does give high utility to the opponent.

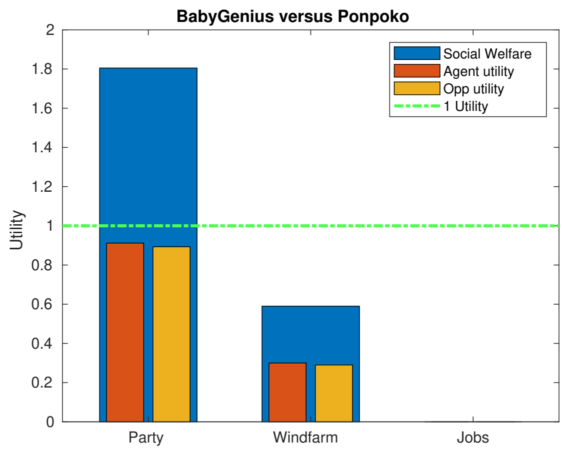

Figure 7: BabyGenius Agent versus Ponpoko Agent on three different domains. Winning Party domain, winning on Windfarm domain if an agreement is obtained, never getting an agreement on Jobs domain

In conclusion it was shown that the agent is able to negotiate properly on different domains and that it wins nearly all negotiations only losing one round out of four to ANAC 2017's winner Ponpoko Agent. On average its obtained utility is lowest for the Jobs domain, a domain with great difference in the possible bid utility near the 1.0 point. The BabyGenius agent therefore performs best on domains with dense bid possibilities.

7 MEDIATED NEGOTIATIONS

Pareto Optimal is by definition a deal where there is no other deal for which the utility of both parties will increase. Having a trusted mediator leading the negotiation would aid in finding such deals, under the condition that the preference profiles of the parties are complete and contain no uncertainties, the preference profiles are fully known to the mediator, and under the condition that the mediator is indeed unbiased. This could then indeed lead to finding an optimal bid on which all parties agree. Furthermore, having a trusted mediator facilitates finding optimal deals the more parties are taking part in the negotiation. Without a mediator, this would become increasingly difficult with every additional party, since every opponent model is prone to errors and every agent has a tendency to adapt their strategies to their opponent's behaviour. This, in turn, makes estimating the opponent models difficult.

In the case that the preference profiles of the negotiating parties are not fully known to the mediator, or the preference profiles contain uncertainties, the mediator will attempt to approach the optimum of the parties. This can be achieved, for example, by implementing a Hill-Climber agent as a mediator. The Hill-Climber accepts a bid if its overall utility is higher than the utility of the most recently accepted bid, thus constantly aiming to increase the overall utility. An alternative to this agent would be the Annealer Agent, inspired by the annealing process of metals. Carried over to the negotiation setting, the Annealer Agent has a tendency to accept individually worse bids at the beginning of a negotiation in order to explore different areas of the utility space, and increase the chance of agents finding optimal bids later on in the negotiation. Nevertheless, neither of these agents can guarantee a Pareto Optimal outcome due to incomplete or inaccurate information. In the case of the Hill-Climber Agent, reaching an optimal outcome might be prevented if the utility of the initial bid is particularly high for one of the negotiating parties. The negotiating party in question, may then not accept any of the following bids, even though those bids might be better for the majority. Furthermore, the starting point of a Hill-Climber in general directly influences the final result and can lead to the agent only finding a local optimum, which may not be on the Pareto Optimal Frontier. In the case of the Annealer Agent, even though the "utility space exploration" reduces the chance of being fixed to a local optimum from the very start of the negotiation, there is still no guarantee that once the agent begins converging to an optimum that it will indeed be a global optimum, i.e. a Pareto Optimal deal.

All things considered, both an optimal and a fair bid may not necessarily be in the best interest of all parties. The main issue lies in determining a fair bid since fairness is often a subjective matter. For one party, a fair outcome could mean the Kalai-Smorodinsky outcome, in other words, an outcome where both parties have the same final utility. For another party, a fair outcome might be equivalent to the Nash product which is defined as $max(U_a \cdot U_o)$. Setting fairness as a requirement for the mediator could thus lead to offers which are not acceptable for certain agents, which could, in turn, increase the number of negotiations where no deal is reached than when there is no mediator in play.

8 FUTURE OUTLOOK

In the current implementation of negotiation agents, the agents are limited to interacting between themselves by making, accepting or rejecting bids, and attempting to estimate the opponent's preferences based off of the bids that the opponent has made. On the topic of the opponent model, a possible extension to how it is estimated could involve allowing the opponent to directly share their preferences. Of course, the preferences that the opponent would share may not be entirely correct, since an agent always aims to get the best offer without having to sacrifice too much in the process. Nevertheless, even a partially correct profile would offer a good reference point for training the opponent model. Based off of the bids that the opponent makes, an agent can validate the preference profile that they had been given, and if necessary update it to fit the observed behaviour of the opponent. Even in the case of a completely incorrect preference profile, if an agent is unable to detect any correlations between the opponent's bids and the given profile, the profile can be treated as a random initialisation of the opponent model and the model can be updated solely on the basis of the bids offered by the opponent.

Another possible improvement involves the communication between agent and user. In the GENIUS interface used over the course of this project, a user is able to set their preferences by setting numerical values as the weights of the relevant issues and as preference of particular values within a given issue. The problem with this kind of approach is that people, in general, struggle with associating precise values with their preferences. Instead, being able to communicate preferences through natural conversation with the agent would facilitate the use of a negotiation platform. [4] Furthermore, it is not uncommon for people to relocate their preferences towards different issues depending on the direction in which a negotiation is going. Taking this behaviour into account, an agent could edit their preference profile - upon confirmation from the user - in order to focus on the now more relevant issues.

Finally, the current implementation resets the opponent model for each run. This approach leads to problems particularly when there is not enough time to carry out the negotiation. Rather than starting from an untrained model every negotiation, one could carry the models from previous negotiations into the new negotiation. Effectively, the agent would learn different opponent profiles and then match these to new opponents. If none of the profiles are a close-enough match, a new profile could be added to the selection which would then be trained be trained as a new model. If the opponent's behaviour matches one of the pre-existing profiles, the agent selects the profile and updates the model over the course of the negotiation in order to attain a more generalised and accurate model for the given profile. The final proposition would require a sophisticated model which would document aspects regarding the opponent's behaviour, based off of which opponents can be profiled.

9 CONCLUSION

In this paper, the strategy and architecture of the BabyGenius negotiation agent are addressed. The acceptance strategy consisting of a not-conceding and a conceding phase, the bidding strategy, consisting of a learning phase, a leverage phase, and a last resort, and the opponent model were addressed and further elaborated with equations and algorithms. The BabyGenius agent is tested on efficiency and robustness where great performance was shown. The BabyGenius agent is able to win every negotiation against the Boulware and Conceder agent. The BabyGenius agent never reaches an agreement with the Hardliner agent which, given the tournament setting, is desired behavior. Testing the BabyGenius agent against the Ponpoko agent (winner of ANAC 2017) shows an average outperformance by the BabyGenius agent. The BabyGenius agent is able to outperform with different utility profiles and on different domains showing the efficiency and robustness of the agent. Under the condition of preference uncertainty, the performance drops, yet does not hamper the overall behaviour of the agent given there are enough bids to base an estimation on. Further improvements on the elicitation method for preference uncertainty would be required to increase the agent's efficiency.

REFERENCES

[1] Robert Axelrod. 1984. The Evolution of Cooperation. (1984).
[2] Tim Baarslag, Koen Hindriks, Mark Hendrikx, Alexander Dirkzwager, and Catholijn Jonker. 2014. Decoupling negotiating agents to explore the space of negotiation strategies. In *Novel Insights in Agent-based Complex Automated Negotiation*. Springer, 61–83.
[3] Réal Carbonneau, Gregory E Kersten, and Rustam Vahidov. 2008. Predicting opponent's moves in electronic negotiations using neural networks. *Expert Systems with Applications* 34, 2 (2008), 1266–1273.
[4] Li Chen and Pearl Pu. 2004. Survey of Preference Elicitation Methods. *Ecole Politechnique Federale de Lausanne, Tech. Rep. IC/2004/67* (2004).
[5] Koen Hindriks and Dmytro Tykhonov. 2008. 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*. International Foundation for Autonomous Agents and Multiagent Systems, 331–338.