Negotiation Assignment: Final Report (Group 30)

Rommy Gobardhan
4124227

Kushal Prakash
5122899

Martijn Straatman
4442504

Jeffrel Hermias
4710088

Mihir Kapadia
5096278

ABSTRACT

In lieu of throwing a nice party with a friend after graduation, a negotiation party is created and realized in order to arrive at a joint decision of how the party should look like. This paper presents the negotiation party created–its structure and various strategies–in order to come up to a certain agreement with the other negotiating party that is as optimally as possible with the author's preference on various utilities. A party performance quantification is done afterwards in order to evaluate the overall performance of the created negotiating party. Subsequently, various points are laid out for future perspectives for this research.

KEYWORDS

negotiation, bidding strategy, acceptance strategy, opponent modeling, Genius, artificial intelligence

ACM Reference Format:
Rommy Gobardhan, Kushal Prakash, Martijn Straatman, Jeffrel Hermias, and Mihir Kapadia. 2019. Negotiation Assignment: Final Report (Group 30). In *Proceedings of IN4010-12: Artificial Intelligence Techniques (TU Delft).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Most entities (whether an individual or organization) negotiate on a regular (even daily) basis. People may not recognize it but negotiations do happen on daily basis. Be it with people close to us like friends and family members, colleagues, people we interact on a day-to-day basis, people are negotiator in different ways. This paper would like to present a negotiation party that the authors devised in order to be used for arriving at a joint decision, as optimally as possible to the author's preference. The joint decision is all about organizing a party and how it will look like in terms of food, drinks, location, cleanup, invitations, and music preferences.

The negotiation session is run in the Genius environment which is programmed in Java. Section 2 discusses the structure and various strategies of the negotiation party. Section 3 provides a detailed description of various strategies modified for this particular negotiation party (Group30party) while Section 4 discusses the party's performance under preference uncertainty. Section 5 highlights the overall negotiation party performance by quantifying the outcomes with respect to other strategies and lastly, Section 6 lays out future perspectives of negotiation strategies in relation to the author's negotiation party.

TU Delft, November 2019, Delft, NL

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

2 NEGOTIATION: PARTY AND STRUCTURE

The section lays out the negotiation party structure modified by the authors. Various strategies are laid out such as bidding strategy followed by the party's acceptance strategy. The opponent model is also touched upon followed by the opponent model strategy.

The negotiation party designed in this research works on the idea of not following the ideal way of "placing only the best bids initially". Instead, a relatively higher utility bid is placed in comparison to the minimum acceptable utility .

Bidding Strategy

The bidding strategy is based on the concept of not placing the best bid initially, under the constraint that it is higher than α. Instead, a bid is generated with an utility between α and 1. In case this bid is not accepted, "selfish" counter bids are produced where the utility of the agent is increased. Since the opponents preference profile is not known, it is hoped that initially these selfish bids are fortunate for both parties. In other words, the counter bid also has an increased utility for the opponent. If the maximum utility is reached in the "selfish" bids without any of them getting accepted, conceding is started till the threshold utility bid is reached.

First Bid: bid with utility closest to $(\alpha + 1)/2$.

- $if\ t < 0.9T$:
  Place bids with increasing utilities till it reaches its maximum (≈ 1) and subsequently bids are placed with decreasing utilities until the utility of the bid is close to α. The next bids have increasing utilities and the

bidding process repeats.

- $if\ t \geq 0.9T$:
  Concede with every bid, until a certain minimum utility is reached (smaller than α).

It is believed this strategy can deceive the opponent into accepting high utility bids which otherwise would not have been accepted.

Acceptance Strategy

For the acceptance strategy, we decided on a combination of different acceptance requirements based on the time, which together formed the following logic equation:

$$AC_{next} \vee AC_{phase1} \vee AC_{phase2} \vee AC_{time}(0.999) \quad (1)$$

The first condition, AC_{next}, will succeed whenever the opponent is giving a bid with a higher utility compared to our own next bid. We chose to include this condition, since if this condition is met, the result will likely be better than if we choose not to accept it, since our own bids will slowly reduce our own utility.
AC_{phase1} is the main acceptance condition of the first phase of the negotiation, which we chose to be the first 90% of the rounds/time. During this phase, the only condition is a constant acceptance condition with a relatively high desired utility of 0.85. This results in the following acceptance condition for this phase:

$$AC_{phase1} = AC_{const}(0.85) \wedge time < 0.9 \quad (2)$$

Phase 2 is the last phase and the acceptance utility here is a function of the time. In this function, we drop the desired utility exponentially to a minimum utility of $U_{min} \approx 0.4$. We chose to maintain a minimum utility to prevent the agent from conceding too quickly in this phase and also to try for a higher resulting utility. The acceptance condition for this face is:

$$AC_{phase2} = AC_{const}(F(time)) \wedge time \geq 0.9 \quad (3)$$

, where $F(time) = 0.4 + 0.45 \cdot cos(15.5 \cdot \text{time} - 13.95)$ (the coefficients 0.4, 0.45, 15.5 and 13.95 are dependent on the chosen α).

The last acceptance condition is the time condition. We included this to make sure that we do not end up with nothing after the negotiation. Since the reservation value is 0, any bid will provide us with a higher utility than no agreement. Our agent does check if the last bid is higher than the reservation value, just in the cases where it might be used.

Using this acceptance strategy, our agent should accept bids with a utility higher than 0.85 in phase 1 and a utility higher than the curve shown in Figure 1 in phase 2.

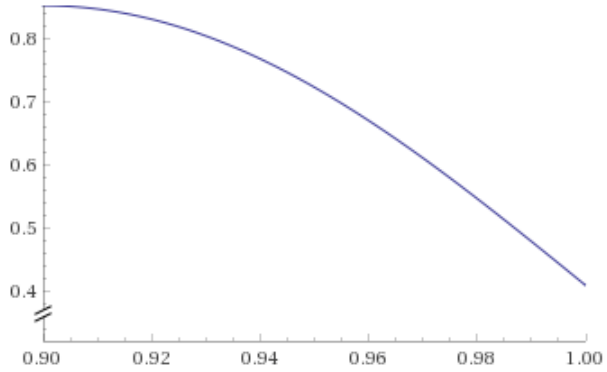

Figure 1: Minimum utility curve for phase 2 of the negotiation

Opponent Model

In the *hardHeaded Frequency Model*, at each step two factors are updated: the issue weight (w_i) and its corresponding value V_j. This update step is performed each time the same value for a particular issue is encountered. It is hypothesized that this model may be inadequate for the long run, under the assumption that bids offered by the opponent at the early stages of the negotiation, better reflect their preferences. As the negotiation process proceeds, it may be possible that the bids are produced with sub optimal preferences (as the opponent starts to generate bids with a lower self utility). In order to alleviate the effect of this phenomenon, a discount factor (DF) is proposed in the current *HardHeaded Frequency Model*, which weigh the values of the initial bids of the opponent higher. As the time progresses, this discount factor is increased:

$$DF = A \cdot (1 - t^2) \quad (4)$$

, where A can be any integer, denoting the possible number of discount steps (currently, $A = 100$). It should be noted that DF is used only to discount the values belonging to the issues which stayed the same in any two subsequent bidding steps. As a result, instead of increasing V_j by a constant value of 1, V_j is increased by DF.

Opponent Model Strategy

As the Offering Strategy (OS) module needs to select the best bid to send out, in terms of the self- and opponents (estimated) utility, the Opponent Model Strategy (OMS) module is consulted. The OS module generates a list of potential random bids with a target self utility. For each of these bids, the self- and opponents utility is determined (self utility U_A and opponents utility U_B). From these, the bid is selected which maximizes the product ($U_A \cdot U_B$).

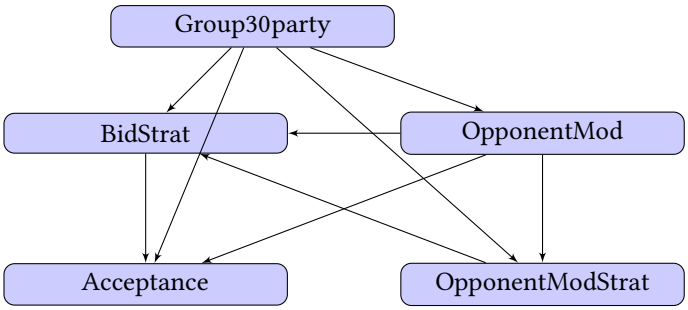

Figure 2: Class dependencies of the Group30party negotiation party.

3 CLASS DESCRIPTIONS

All the classes are implemented using BOA framework which consists of four components: Offering strategy, Acceptance strategy, Opponent model and opponent model strategy.

Group30party

class extends BoaParty and initializes the Offering strategy, Acceptance strategy, Opponent model, Opponent model strategy and all the components required to run the party.

BidStrat

class extends OfferingStrategy and contains functions implementing the Bidding strategy. **nextBidUtil()** method computes the next bid using the strategy defined earlier and **determineNextBid()** returns the next bid computed.

Acceptance

class extends AcceptanceStrategy and contains functions implementing the acceptance strategy. **determineAcceptability()** method checks for the acceptability of the bid received from the opponent.

OpponentMod

class extends OpponentModel and contains function implementing the opponent model. **updateModel()** updates the estimated weights and values of the opponent using the current bid of the opponent.

OpponentModStrat

class extends OMStrategy and contains functions implementing the opponent model strategy. **getBid()** method returns the best bid for the opponent.

Figure 2 shows the dependencies betweeen the classes.

4 PREFERENCE UNCERTAINTY

In order to let the agent perform well under preference uncertainty, the weights of each issue and its corresponding values have been determined by adapting similar techniques used in the hard headed frequency opponent model:

- **Issue weights:** As in the case of the opponent model, in each bid of the given bid ranking, the frequency is computed at which certain issues remained the same in subsequent bids. At any given bid in the bid ranking, if the value of an issue did not change compared to the previous bid, it is assumed that this issue is more important. After determining the counts for each issue, using the sum of the counts of all issues, each weight has been normalized. However, this method may pose the following drawback. If there are two values in an issue alternating in subsequent bids, these two issues might be classified as unimportant, while the agent may actually be indifferent to both values for that particular issue.
- **Value weights:** In this case, it is important to identify the relative importance of values for each issue. It is assumed that preferred values for each issue is better reflected by the bids ranked at the top (a higher ranking means a higher utility). To include this assumption, only the top 10% of the bid ranking is used to estimate the weights of the values for each issue. To this end, the normalized frequency of each value is used as the weight.

For this particular case, the preference uncertainty has been implemented in the **boa party** class, consisting of the following methods:

- **estimateUtilitySpace()** : As instructed in the user guide [1], this method can be overridden to adjust the utility space under uncertain preference conditions.
- **updateDifference()** : This method takes as input a list of issues, a dictionary containing issue frequencies and two subsequent occurring bids in the bid ranking. If the value of an issues in both bids are the same, the frequency of the unchanged issues are updated in the dictionary.

In order to test the behaviour of the agent under preference uncertainty, two round based negotiation sessions were carried out against the Boulware agent. Table 1 presents the simulation parameters with the corresponding results for both scenarios. As expected, in the case where the preference profile is known, the Boulware agent concedes when the deadline approaches. As a result, a bid is accepted by the opposing party with a lower utility. However, in the case of preference uncertainty it can be seen that the agent accepts the opposing bid immediately where the utility of the Boulware agent is 1. Based on the acceptance conditions

[1] https://tracinsy.ewi.tudelft.nl/pub/svn/Genius/doc/userguide.pdf

described in Section 2, it is likely that the opponent offered a bid with an acceptable self utility.

5 PARTY PERFORMANCE QUANTIFICATION

To see how the negotiation party performs, several tests have been run. First, the performance of the party against different parties in the *Party* Domain will be checked and then the performance of the party against itself in other domains will be check to see how generic the party is.

Performance of the negotiation party in the *Party* Domain

To test the negotiation party in the *Party* Domain, multiple negotiation sessions will be run where the party will negotiate against itself, a Boulware party, a Linear Conceder party and a Hardliner party. Against each of these parties, 20 sessions will be run, each with different(random) utility profiles and a timer of 30 seconds. The results will be grouped per party in a table and then discussed shortly.

Performance against itself. In Figure 3 the number of utility yields can be seen. These results were created by running 20 negotiation sessions with different utility profiles for the parties. Since the parties are the same, it has no use to split the results in these case other than to check for differences in being the first bidder or the second bidder.

The average utility of all the perceived utilities is 0.713. This is a fine score, but it can definitely be improved. Only one of the negotiation sessions took longer than 5 rounds of bidding, because the agents accept a bid too soon when negotiating against each other. This amount of round is extremely low considering that before the deadline of 30 seconds multiple thousands of rounds could have been performed if no bid would have been accepted. This can be explained by the combination of the bidding strategy and the acceptance strategy. Where the bidding strategy starts with a relatively low utility for the agent itself and slowly increases his own utility, the acceptance strategy will make the agents accept a bid if it is higher than its next bid. However, since both the agents start out pretty low with their bids, it is very likely that 1 agents proposes a bid that will be higher than the other agents next bid resulting in a low number of rounds. This also prevents the agents from proposing bids with a high utility for itself in this case. Therefore, a Pareto optimal result was only achieved once and there was no Nash solution. The few high utilities that were perceived were caused by coincidence between the profiles, one party gave a slightly higher utility for itself compared to the last bid and the random choice for a bid with that utility resulted in a very high utility for the other party. If the negotiation sessions were to be longer, this could have been accredited to the opponent model and opponent model strategy, but since only a few rounds have been played the model was not accurate yet.

This party will probably perform better against itself if the AC_{next} acceptance condition is removed or if the bidding strategy is completely overhauled.

Performance against Boulware agent. For testing the performance against the Boulware against, 10 negotiation sessions were run with the Boulware agent as the first bidder and 10 with the Group30Party as the first bidder.

As can be seen in Figure 4, the Boulware party performs significantly better in general. The Boulware party got a higher result than the Group30 party in every single negotiation session although the difference was close to 0 in some sessions. The average utility yield of the Group30 party is 0.722 and of the Boulware party 0.957. The average utility is a bit higher than the average utility when the party negotiates against itself, but when comparing to the Boulware party, the party does not perform well enough yet. The difference in utility seems to be caused again by the acceptance condition AC_{next} or the bidding strategy as a whole. The Boulware agent always starts with the bid that yields itself the maximum utility and in about half of the cases, that bid was good enough for the Group30 party, since it was starting out with a lower bid than the bid of Boulware would yield effectively ending the negotiation session immediately. If the first bid of the Group30 party would be of higher utility to itself, the AC_{next} condition would not be met as much on the first bid. This would cause the negotiation to continue to a point where the Boulware would slowly concede allowing the Group30 party to receive a better deal. Therefore, the AC_{next} conditino should be removed or the initial bids should give a higher utility for the performance to increase.

A positive thing in these negotiation sessions was that a Pareto optimal outcome was reached in all but 1 session meaning that almost no bid was accepted where both parties could have done better. A Nash outcome was reached in 20% of the sessions, which is also better than when the party performed against itself.

Performance against Linear Conceder agent. The Group30 agent performed very similar against the Linear Conceder agent when comparing it to the Boulware agent as can be seen in Figure 5.

The average utility yield is 0.722 and 0.952 for the Group30 and the Linear Conceder agents respectively. These are both extremely close to the average yields of the negotiation sessions with the Boulware agent. This is caused due to the Boulware and the Linear Conceder agents starting sessions in the same manner, which is starting with the highest utility bid for themselves. This gives the similar issue that in some sessions, the Group30 agent accepts a bid very early while it would have yielded a higher utility by waiting some more.

Table 1: Bid utility for the developed agent versus the Boulware agent under known preferences and uncertain preference conditions.

	Preference Certainty		Preference Uncertainty	
	Agent Group 30	**Agent Boulware**	**Agent Group 30**	**Agent Boulware**
Size Bid Ranking	-	-	50	-
Rounds	60	60	60	60
Agreement round	55	55	2	2
Utility	0.77	0.47	0.66	1

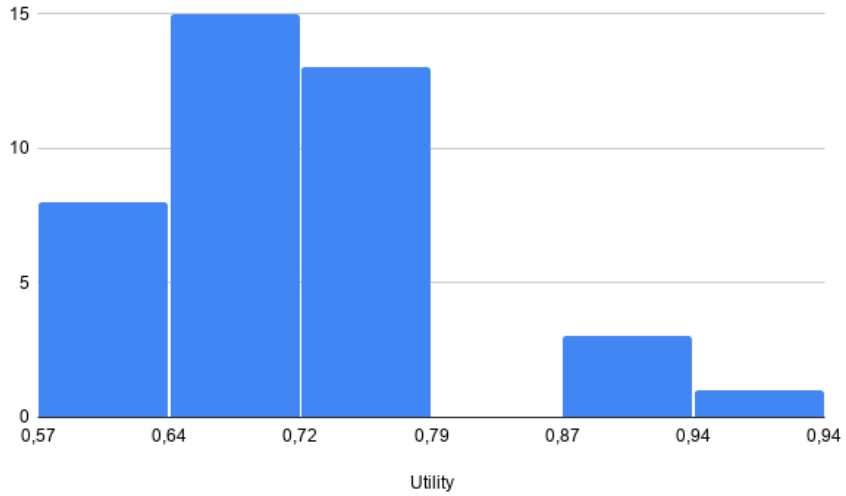

Figure 3: Utility yields from negotiating against itself in 20 different sessions

The number of Pareto optimal and Nash outcomes is also similar to that of the Boulware party further showing that those yield similar results against the Group30 agent. Again, these results should improve significantly by removing the AC_{next} condition or increasing the utility on the starting bids.

Performance against Hardliner agent. The Hardliner agent completely dominates the Group30 agent as can be seen in Figure 6.

The Hardliner agent yields a perfect utility in every negotiation session and the average of the Group30 agent is the lowest of all tests, which is 0.612.

This domination is caused by the Hardliner agent refusing to ever concede on the maximum utility for itself, while the Group30 agent tries to get the best bid for both over the whole negotiation. However, if no time is left Group30 agent will accept any bid just so its utility is not 0.

No real improvement on this seems to be possible for the Group30 agent, since the Hardliner will never concede just a little even if it means that the Group30 agent improves a lot. Because the Hardliner agent always yields a utility of 1, every session was a Pareto optimal outcome and there was 1 Nash outcome as well.

Performance of the negotiation party in other domains

To see how generic the Group30 party is, it will be tested in negotiation session in other domains rather than the *Party* domain. These will then be compared(including the comparison in the *Party* domain of the last subsection) to determine how generic the agent is. The domains chosen for this analysis are the *Car* domain and the *Pie* domain. For both of these, a negotiation session will be run between the Group30 agent and itself, the Boulware agent, the Conceder agent and the Hardliner agent(for every session in a domain the same utility profiles will be used).

Car domain performance. In the *Car* domain negotiation session the agent performs worse than in the average negotiation in the party domain as can be seen in Table 2.

The opponents do also performs worse than their average however(except for the Hardliner). Therefore, this scenario is probably a scenario with more outlying wishes in the utility

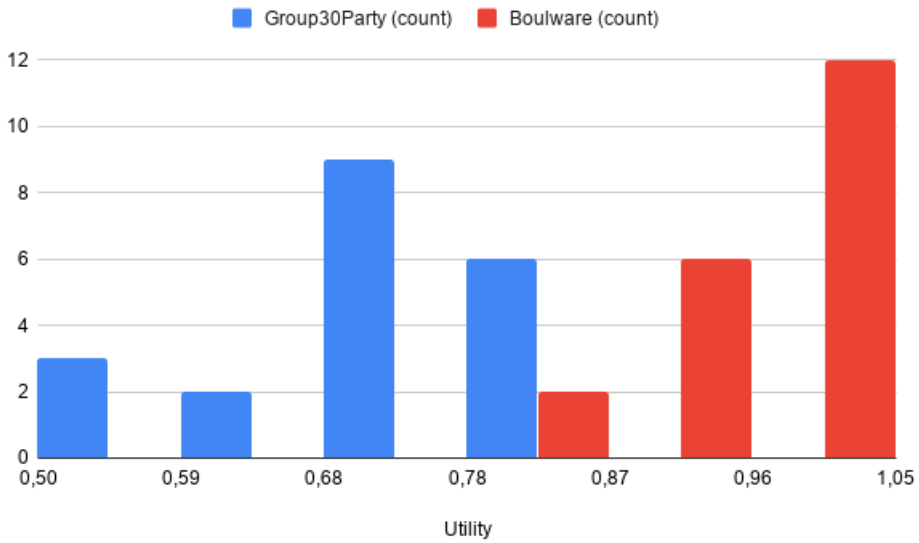

Figure 4: Utility yields of the Group30 agent and the Boulware agent

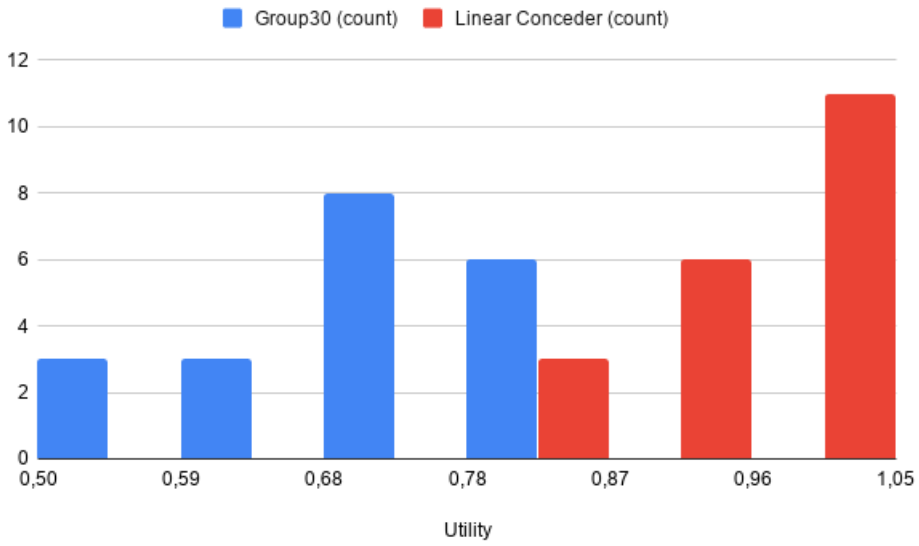

Figure 5: Utility yields of the Group30 agent and the Linear Conceder agent

profiles causing it to be hard to obtain high utility values for both parties. This is especially the case since Pareto optimal outcomes have been reached in every negotiation except for with the agent itself, meaning it could not have ended better for both parties. No Nash outcomes have been reached though.

Since the Pareto optimal outcomes have been reached in multiple of the sessions, it can be expected that in other combinations of profiles Pareto outcomes can also be found. The chance for this will increase upon implementing the improvement on the agents as specified in the performance analysis on the party domain.

Holiday domain performance. In the *Holiday* domain, the agent performs around average and sometime slightly better. This is probably due to the way the domain is constructed

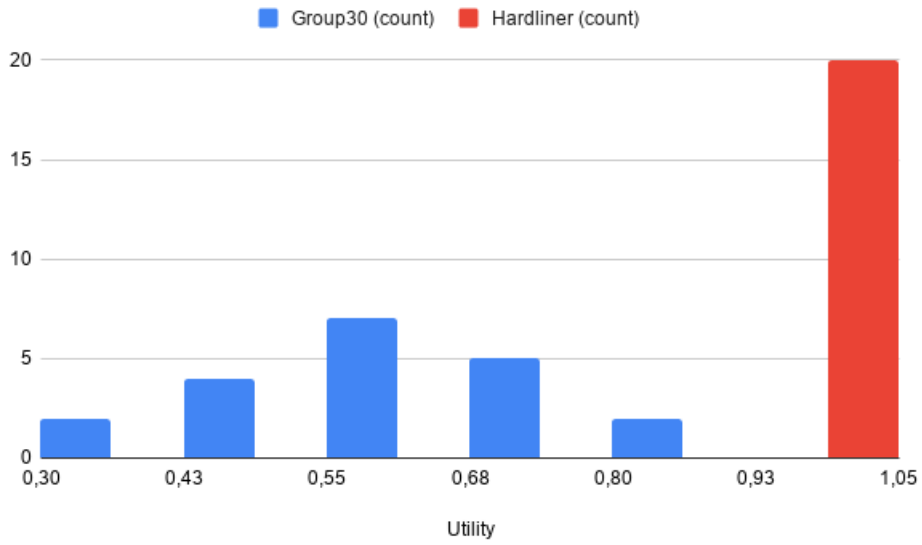

Figure 6: Utility yields of the Group30 agent and the Hardliner agent

Table 2: Utility yield in a negotiation session in the *Car* domain of Group30 vs itself, Boulware, Conceder and Hardliner agents

	Group30	Opponent
Group30	0.605	0.585
Boulware	0.624	0.898
Conceder	0.624	0.898
Hardliner	0.529	1

as was the case with the worse outcomes in the *Car* domain. The results can be found in Table 3.

Table 3: Utility yield in a negotiation session in the *Holiday* domain of Group30 vs itself, Boulware, Conceder and Hardliner agents

	Group30	Opponent
Group30	0.647	0.646
Boulware	0.799	1
Conceder	0.799	1
Hardliner	0.799	1

When not negotiating against itself, only Pareto outcomes have been reached indicating that this agent will perform fine in this domain.

6 FUTURE PERSPECTIVES

This section discusses some future perspectives on using a mediator and extending capabilities of the parties in the negotiation environment. The following subsections highlight these two food for thought.

Using a mediator

When a trusted mediator is put into the negotiation picture, a fair outcome will emerge due to the fact that preferences of each party are kept in private by this arbitrator. By keeping these preferences in secret, other parties cannot perform opponent modeling (and opponent modeling strategy) which allows a fair-bidding game based solely in their own personal goal. Arriving to an optimal outcome might take a while since each party has to (re)adjust their bidding and acceptance strategies from time to time since they are unable to model their opponent properly. They will only know the outcome but not their opponent's preferences. The best thing a negotiation party can do is to bid according to its own choice & preferences and see it from there.

Capabilities

The Genius negotiation environment is built to explore multiple negotiation domains. Firstly, within any negotiation session, the measure of the efficiency of a negotiation strategy depends highly on the knowledge of opponent's strategy. Efficient modelling of an opponent is key to forming a strong negotiating agent. Secondly, the efficiency of a negotiation strategy of any agent would also depend on the domain

knowledge as well as preference profiles. While optimising the agent to be as generic as possible while fulfilling a utility maximising criterion, the preference profile may extremely difficult to formulate. This is further affected by the sensitivity of the agent towards specific changes of the opponent's behaviour. Thus, an agent may perform extremely well in a case where the domain information is complete and known, but under-perform in cases where domain has unpredictable issues. Further, the domain unpredictability affects both the agent as well as the opponent. Thus, information or lack of it there-of, is equally affecting to either of the agents negotiating. Given the challenges and dependencies above, it can be seen that there could not be possible, any additional capabilities which would increase the utility of the party.