Agent21 - A Similarity Trade-off and Opponent Modelling Approach to Bilateral Negotiation in Complex Environments

Keniel Peart (33332495), Surpitha Konaje (32864477), Sowmya Buddharaju (33252629), Parisha Rafiq Ullah (33119082)
Group : 21
Word Count : 2500

Abstract—**In this paper, we present a negotiation strategy that combines Trade-Off, Time-based concession and Frequency Opponent modelling. The goal of this work is to demonstrate by experimental analysis that the combination of these strategies may improve agent negotiation and in return present satisfactory results.**

I. Introduction

Negotiation is the process by which agents communicate to achieve a result in their own best interest. Negotiation is employed in various scenarios, including product purchases, personal conflicts and game theory. Our agent participates in a negotiation tournament that follows the Stacked Alternating Offers Protocol; one agent makes an offer, and the other assesses if the offer meets its requirements. If not, the other agent makes a counter offer. This process continues until there is an agreement or time runs out. The tournament had three main domains - SportHal, Party and Windfarm, which were of small, medium and large size respectively.

II. Literature Review

In this section, we will be discussing the background of strategies chosen for Agent21.

The performance metrics of the tournament are the agent's average utility, average distance to Nash Bargaining Point and the total number of agreements. Based on these metrics, we focused on particular literature to develop Agent21.

The work done in [1] highlights the various issues that agents face while negotiating. Based on this work, we considered opponent modelling. Opponent Modelling uses insight from the negotiation space to infer the opponents' preferences, allowing us to create bids that are not only suitable to our preferences, but to the opponent's as well. A prominent approach is the Jonny Black algorithm [2]. The Jonny Black algorithm keeps track of the frequency of the opponent's bids and uses that to ascertain their most preferred offer. By considering the opponent's preferences, we can reduce the distance of our agreements to the Nash Bargaining point.

While negotiating, the agent has limited information about the preferences of the opponent, making it difficult to reach an agreement. This directly impacts the agent's utility. To overcome these issues, the concept of similarity used in the trade-off strategy allows the agent to represent the domain of negotiation issues. The fuzzy similarity enables us to cope with uncertainties in our opponent while still coming up with offers that might entice them. This strategy also explores all possible outcomes in the domain, thus increasing our likelihood of reaching an agreement as well as obtaining higher utility [3].

III. Negotiation Strategy

A. Agent21

In this section, we discuss the negotiation strategy of Agent21 in detail. For simplicity, we will refer to Agent21 as AgentA and its opponent as AgentB. AgentA's negotiation model consists of 3 main components. The first component is the Offering Strategy which dictates how AgentA generates an offer at time t. The second component is the Opponent Model, which tries to plot the opponent's preferences based on AgentB's previous offers. The last component is the Acceptance Strategy, which determines whether AgentA should accept or reject an offer. Given a negotiation outcome and an acceptance threshold at time t, the component returns a boolean indicating whether to accept or reject the offer. An overview of Agent21 is given in Algorithm 1. The individual steps of Algorithm 1 are explained in the following.

Algorithm 1 : Upon receiving a counteroffer from AgentB, AgentA records the new bid using its opponent model component. The agent then decides whether to make a counteroffer or accept AgentB's bid. If the bid is deemed acceptable by the acceptance strategy, the bid is accepted; otherwise, a new bid is generated using AgentB's bid as input.

Algorithm 1: The overall flow of Agent21

```
Data: $t^c$ // current time
      $t^{max}$ // max time
Result: $Action$ // Accept, End, Offer
while $t^c < t^{max}$ do
    $B_o \Leftarrow recieveMessage$
    $recordsBids(B_o)$
    if $B_o$ is null then
        $Offer(proposeInitialBid())$ ;
    else
        if $isAcceptable(B_o, t^c)$ then
            $Accept(y^t)$;
        else
            $Offer(generateBid(B_o))$ ;
        end
    end
end
```

1) Offering Strategy: The offering strategy employed by AgentA aims to produce bids that return an aspirational utility for AgentA and are similar to AgentB's last bid. The offering strategy corresponds to line 11 in Algorithm 1. When receiving a new bid from AgentB at time t_c, the agent generates a list of offers that match the utility of AgentA's last utility and from that list finds the bid that is most similar to AgentB's last bid. This bid is determined by comparing each selected issue option to AgentB's bid.

The general idea behind the offering strategy realized by AgentA is to apply the similarity criteria principle to make trade-offs in negotiations [3]. In more detail, Trade-Off is a mechanism that seeks to find a bid that has the same utility (aspiration value) as its previous bid but which may be more desirable (have a higher utility) for its opponent. One issue that is presented by this algorithm is that it requires the Agent to select a bid that is likely to improve the utility of the opponent without knowing its opponent preferences. To accomplish this feat AgentA must be able to do the following:

1) Generate a list of all bids with the same utility as AgentA's last bid x (aspiration value).
2) Select from the list a bid (y) that AgentA believes will be preferable to AgentB than x. AgentA would select the most similar bid to AgentB's last bid.

The cornerstone of this algorithm is to choose a bid that is most similar to AgentB's last bid. The similarity metric is resolved by the following two definitions.

1) The similarity between two bids x and y over a set of issues I is defined as:

$$Sim(x, y) = \sum_{i \in I} w_i^a . Sim(x_i, y_i)$$

where

$$\sum_{i \in I} w_i^a = 1$$

and Sim_j is the similarity function for issue i over the domain D

2) The similarity between two issues x_i and y_i is defined as:

$$Sim_j(x_j, y_j) = \sum_{1 \le t \le m} w_t . (h_t(x_i) \leftrightarrow h_t(y_i))$$

where w_i is a set of weights and h_t is the evaluation function of the issue option

The proposed trade-off approach is implemented in Algorithm 2. The algorithm is $\mathcal{O}(n)$ and takes as input AgentB's last bid Bid_b, AgentA's aspirational value θ (Utility of AgentA's last bid) and the list of all bids in the domain B. The output of the algorithm is the counter-bid proposed after trade-off.

In algorithm 2, first, the utility of the current indexed bid is calculated (line 8). Then a check is done to see if the bid utility lies in a close range to that of the aspirational value (line 9). If that is the case, the bid is compared with AgentB's last bid to generate the similarity score (line 10). Then the similarity is checked with a value lambda which represents the maximum similarity value (line 12), and the value of the bid is assigned to the trade-off result (line 13). This process continues until all bids in the bid space are checked, resulting in the most similar bid being returned. If the trade-off bid is null at the end of going through all the bids, the last bid in the space is returned (line 17-18)

Algorithm 2: Trade-Off Algorithm

```
   Data: $Bid_b$ // AgentB last bid
1        $\theta$ // AgentA aspirational value
2        $B(b_1, b_2, b_3 ... b_n)$ // Domain bid space
   Result: $Bid_{trade-off}$ // Bid returned from
           trade-off process
3  $bd := null$;
4  $tempBid := null$;
5  $lambda := -1.0$;
6  foreach $b_i \in B$ do
7      $tempBid := b_i$
8      $tempUtility := getUtility(tempBid)$
9      if $|tempUtility - \theta| < .01$ then
10         $similarity :=$
           $determineSimilarity(tempBid, Bid_b, B)$
11         if $similarity > lambda$ then
12             $lambda := similarity$
13             $bd := tempBid$
14         end
15     end
16 end
17 if $bd == null$ then
18     return $tempBid$;
19 end
20 return $bd$;
```

a) Concession Strategy: The trade-off algorithm is advantageous because it searches the bid space for a bid that maintains AgentA's aspiration value; our utility gain never decreases. However, because of this advantage, the chances of an agreement without concession is unlikely. The concession strategy is hence defined as the mechanism used to decrease the current aspiration value to increase the chances of an agreement. This strategy is time-dependent [1], and as a result, as time goes by, AgentA concedes more on its aspiration value. The concession strategy is defined in Algorithm 3.

In algorithm 3, alpha is a function depending on time and parametrised by a value c. Where c defines the value amount, the agent is likely to concede each round.

Algorithm 3: Concession Strategy Algorithm

```
Data: $min$ // minimum utility
1       $max$ // maximum utility
2       $c$ // concession rate
3       $i$ // initial concession
Result: $targetUtility$ // new aspiration
        value
4 $alpha := i + ((1 - i) * t^{\frac{1}{c}}$;
5 $target := min + (1 - alpha) * (max - min)$;
6 return $target$;
```

2) Opponent Model: Opponent modelling realized by AgentA aims at predicting the weights on issues and the preferences ordering of the options for every issue. To accomplish this, we assume the following :

1) More preferred options appear more frequently in bids.
2) It is less likely for the negotiating opponent to change from its best option to another.

The process of opponent modelling corresponds to line 4 in Algorithm 1. When receiving a new bid from AgentB at the time t_c, the agent records the count of each option as they appear in the bid. Then, the preference ordering for options is calculated by giving the most frequent option one while the remaining option as their $\frac{rank}{number of options}$ and the weights are calculated by normalizing the frequency values. The main idea behind this mechanism is drawn from the Johnny Black paper [2].

a) Nash Utility: The nash product is a metric used to evaluate the fairness of bilateral negotiations. As a result, we designed a component to minimize the distance of the offers accepted by AgentA from the nash point. Furthermore, the opponent modelling component allows AgentA to improve its prediction of AgentB's preferences, thus improving the chance of assuming the nash point. The nash point algorithm is modelled in Algorithm 4. In algorithm 4 a check is done to determine if the new data captured by the opponent model updated the bid space (line 6). There is a method called whenever new data is added to the frequency table to update this boolean. Then each bid in the space is added to the Pareto line. This is done by comparing the current bid with all the bids currently on the line to determine whether there is a better bid for an agent; if that's the case, the bid is not added to the line; otherwise, the bid is added. Also, the method goes triggers a function that goes through the line and removes any bid which should no longer be on the line (line 8). After that is done, the nash product of each bid on the Pareto line is calculated, resulting in the bid with the highest product becoming the nash bid/ point (lines 10 - 14).

Algorithm 4: Nash Utility - Generate Nash Point

```
Data: $updated$ // boolean to check if the
      opponent predicted utility was
      updated
1     $B(b_1, b_2, b_3...b_n)$ // Domain bid space
Result: $nashBid$ // Nash Bid/point
        returned
2 $max := -1.0$;
3 $product := 0$;
4 $nashBid := null$;
5 if $updated$ then
6     foreach $b_i \in B$ do
7         $addToPareto(b_i)$
8     end
9     foreach $p_i \in paretoline$ do
10        $product := p_i.getAgentUtility() *$
          $p_i.getOpponentUtility()$
11        if $product > max$ then
12            $nashBid := p_i$
13            $max := product$
14        end
15    end
16 end
17 return $nashBid$;
```

3) Acceptance Strategy: This stage corresponds to lines 8–12 in Algorithm 1. When the target utility u_t has been determined, the agent needs to examine whether the utility of the latest counteroffer B_o is better than u_t or close to the nash point, i.e. the distance to nash is inclusively between 0 and .14. If either of these two conditions is satisfied, the AgentA accepts this counteroffer and finishes the current negotiation session. If, however, time is .9 or above, the agent considers the following:

1) If time is greater than .9 and the utility of the counteroffer is greater than AgentA's minimum utility; Accept the offer.
2) If time is greater than .95; Accept the offer. AgentA seeks to prioritize an Agreement over a disagreement even though its received utility may be low

Otherwise, the agent constructs a new offer using the trade-off method in the offering strategy with a target utility determined by the Concession Strategy.

IV. Evaluation

Figure 1 shows Agent21's performance against the Boulware agent, which is hardheaded. In contrast, Figure 2 shows

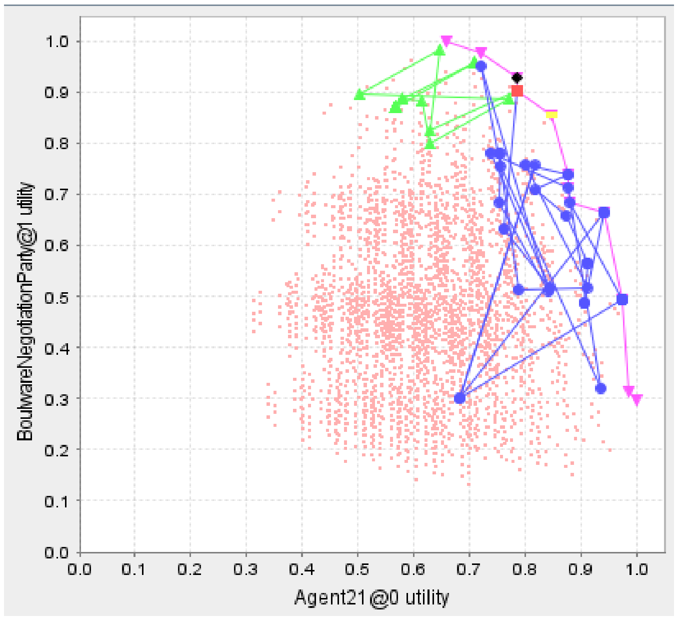

Fig. 1. Negotiation space from Genius showing bids offered by Agent21 in blue, and Boulware agent in green. The agreement is in red, the NBS is in black.

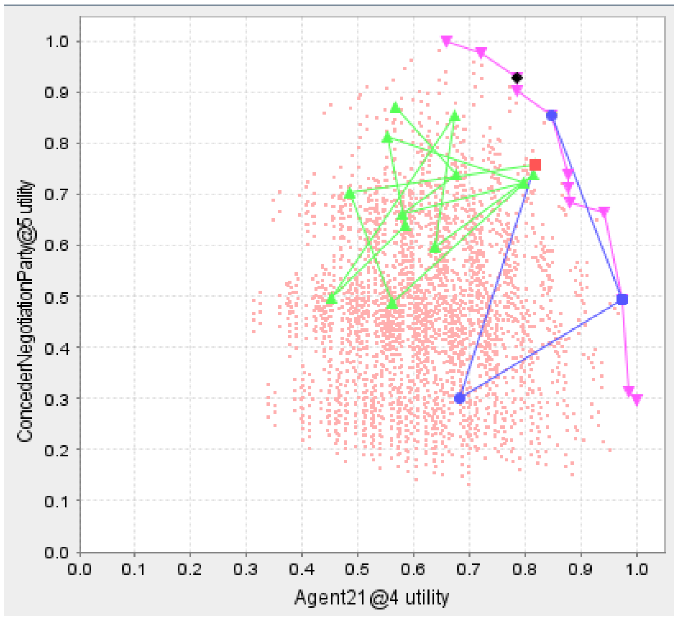

Fig. 2. Negotiation space from Genius showing bids offered by Agent21 in blue, and Conceder agent in green. The agreement is in red, the NBS is in black.

Agent21's performance against the Conceder agent. For comparison, both negotiations were run on the same domain with the same preference profiles. Agent21 achieved roughly the same utility in both negotiations (0.8), implying that our agent is not weak to either strategy, selfishness or generosity, as it has performed similarly against both. It should be noted that in both negotiations, our agent offered the bid that the other party accepted. This shows that our bid offering strategy can accommodate the other agent and find a good offer for both parties.

TABLE I
PERFORMANCE OF AGENT21 AGAINST A SELECTION OF AGENTS FROM THE GENIUS PLATFORM

Domain Name	Domain Size	Dist. to Nash	Utility
Laptop	Small	0.1309	0.843170419
Party	Medium	0.1210	0.959598837
WindFarm	Large	0.4199	0.814430246

We ran tournaments on Genius on variously sized domains against agents Jonny Black, AgentM, and Bayesian Agent to evaluate our agent. The results of these tournaments are shown in Table I.

Figures 3 and 4 show our agent's performance in a much larger tournament against 30 others. Figure 3 shows Agent21's average utility on each domain, alongside the median utility obtained by all agents on the same domain. Figure 4 does something similar for the average distance from the Nash point. Table II describes the different domains involved.

The Nash Bargaining Solution is obtained by maximizing the product of the utilities minus the reservation value. The

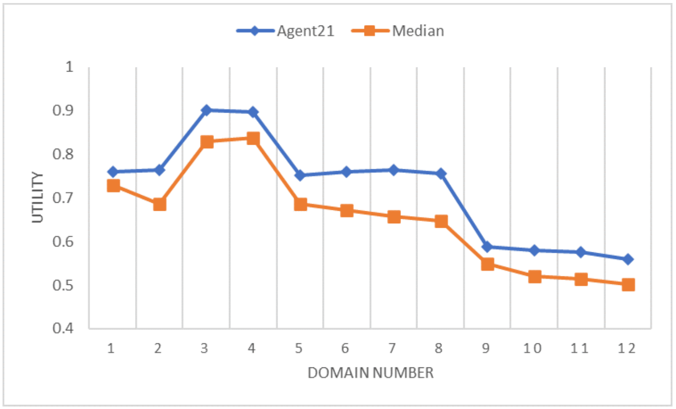

Fig. 3. Utility achieved across all domains, higher is better.

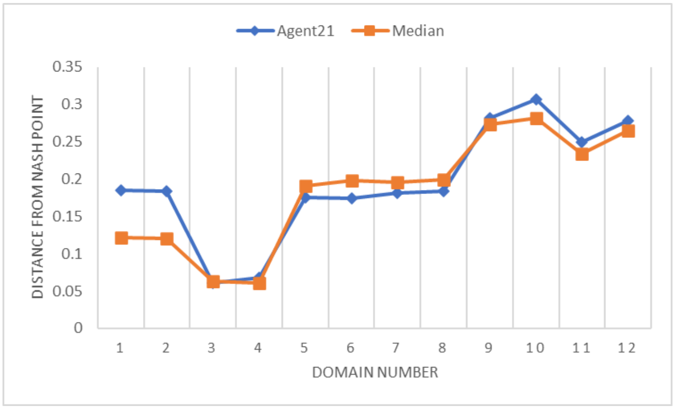

Fig. 4. Distance from Nash point across all domains, lower is better.

TABLE II
DESCRIPTION OF DOMAINS INCLUDED IN THE TOURNAMENT; DOMAINS AND THEIR VARIATIONS HAVE BEEN GROUPED.

Domain No.	Domain Name	Domain Size	Possible Offers
1-4	SportHal	Small	243
5-8	Party	Medium	3072
9-12	WindFarm	Large	7200

distance from the Nash point is used as a metric for evaluating the agent's performance because it shows not just how well the agent itself came off but also how well the opponent has done. The intent is to make both parties happy.

From figure 3 we can see our agent consistently did better in terms of utility than the median value. The utility across domains 1 to 8 are consistent with what we achieved in our own tournaments on small and medium domains on Genius, showing that our agent is performing optimally. However, across domains 9 to 12, we can see a sharp drop in the utility compared to domains 1 to 8. Thus our agent performed suboptimally when negotiating on a larger domain, and this is an area that we could improve on. It should be noted that the median utility for all agents followed the same trend and was lower on the larger domains.

Figure 4 shows that except for the first 2 domains, our agent was very close to the median distance from the NBS for all domains. For the medium-sized domains, we did slightly better; for the large domains, we did slightly worse. Agent21 performed significantly worse on the first 2 domains, which were small domains with a low number of known offers. Additionally, in both our tournament and the ones against other agents, the distance from Nash was much worse on larger domains. This can be improved by incorporating preference elicitation into the agent. In doing so, the agent would have a better idea of its own utilities and thus calculate a more accurate Nash point. Furthermore, instead of using bid similarity to find and offer bids, we could use the weights in our opponent model to find an enticing bid for the opponent.

The performance of Agent21 could be improved by reducing the number of times we update our opponent model. The agent currently updates the opponent model every time an offer is received, which slows down the agent's performance. Further work needs to be done to find the right interval between updates. Currently, the agent searches the entire bid space for bids similar to the opponent's offer. This increases the time we take to offer a bid and reduces the overall number of offers in the negotiation, leading us to perform poorly in larger domains. This can be seen in our agent's performance in the WindFarm domain, the tournament's largest domain. To overcome this, we could implement search strategies or genetic algorithms to find similar bids faster. We could also consider situations where it might be beneficial to disengage from the trade-off strategy altogether. Conversely, trade-off might show poor performance on significantly small domains such as the laptop domain provided by the GENIUS platform. With only three issues and three options per issue, very few bids can be similar to the opponent's offer.

V. CONCLUSION

We provide an overview of the methods followed by Agent21 to achieve a good performance in a tournament against other agents from the same module. The agent was designed to integrate the three tactics of trade-off, time-based concession, and opponent modelling.

In the tournament, Agent21 performed satisfactorily. During the evaluation of the agent, we discovered that the agent performs better on small and medium domains. To improve our performance on larger domains, we can reduce the search space to find similar bids. We can also preference elicitation to update the weights of the trade-off approach.

REFERENCES

[1] P. Faratin, C. Sierra, and N. R. Jennings, "Negotiation decision functions for autonomous agents," Robotics and Autonomous Systems, vol. 24, no. 3-4, pp. 159–182, 1998.
[2] Yucel, O., Hoffman, J. and Sen, S., 2017. Jonny Black: A Mediating Approach to Multilateral Negotiations.
[3] Faratin, P., Sierra, C. and Jennings, N., 2013. Using similarity criteria to make issue trade-offs in automated negotiations.