CS451 Introduction to Artificial Intelligence
Project Final Report

Group 5 (Tuğçe Özgirgin - Kerem Yıldırır - Alkın Korkut)

I. Introduction

In our project, we developed an AI negotiating agent explicitly tailored for the ANAC2024 tournament conditions, aligning with the project requirements of the CS451 Introduction to AI course. This final report is a comprehensive documentation of our negotiation agent's development, strategies, and performance. Our agent is structured into three main components: bidding, acceptance, and opponent modeling, which will be explicitly discussed in the upcoming parts.

II. High-Level Description of the Agent

Our agent uses a threshold-based acceptance strategy which dynamically calculates and sets the threshold value over negotiation time and based on the opponent's behavior. More specifically, the threshold value for accepting offers is dynamically adjusted based on the opponent's concession rate and the remaining rounds in the negotiation. For the bidding strategy, at the very beginning within a specified time frame, our agent will typically follow less concessive behavior by making a little concessions. After that, our agent adapts its bidding strategy based on the behavior of the opponent's movements and the progress of the negotiation. Furthermore, our agent uses a Bayesian modeling approach to estimate the opponent's reservation value. Our agent can estimate the opponent's reservation value, by keeping track of the opponent agent's utilities for the offers made during the negotiation and applying the Bayes model. Then this opponent's reservation value can be used while the negotiation continues.

III. Components

1. Acceptance Strategy

The acceptance strategy employed by our negotiating agent relies on a threshold-based approach, where offers are assessed against a dynamically adjusted threshold value. This threshold diminishes over time, reflecting the decreasing value of time and escalating pressure to secure an agreement. To enhance control over the adjustment process, we introduced a method that calculates the current threshold using exponential decay. By adjusting the exponential decay rate based on the negotiation progress, we ensured smoother transitions and reduced the risk of sudden changes adversely affecting negotiation outcomes. The mentioned threshold adjustment solely occurs if the opponent's concession rate surpasses 0.005.

In this case, the current threshold is recalculated using the exponential decay method. The existing threshold remains unchanged if the opponent's concession rate does not meet this criterion. The initial threshold is calculated based on 150% of the utility of the Nash product bid. This ensures that the initial threshold is set at a level that incentivizes accepting offers with the potential for mutual benefit, optimizing the negotiation process across different scenarios.

The acceptance conditions include a check for the utility of the current offer relative to our agent's previous bid. If the utility of the current offer exceeds our agent's previous bid and the negotiation has progressed beyond the initial step, the offer is accepted. This adjustment enhances the agent's flexibility in negotiating mutually beneficial agreements. Additionally, we adapt acceptance conditions based on the remaining negotiation rounds. If fewer than two rounds remain and the utility of the offer surpasses the reserved value, the offer is accepted to lower the risk of failing to reach an agreement within the allotted time frame.

The final implementation of the acceptance strategy differs from the design report in several key aspects, reflecting adjustments made to enhance efficiency and adaptability.

1. Nash Product Bid Calculation Function:
    - The latest iteration introduces a new function that identifies the Nash product. This function aims to select bids that offer the maximum potential mutual utility, thereby facilitating agreements that benefit both parties optimally.
2. Refinement of Initial Threshold Calculation:
    - The acceptance strategy now determines the initial threshold based on 150% of the utility of the Nash product bid. This adjustment ensures that the initial threshold is set at a level that incentivizes accepting offers with the potential for mutual benefit. By incorporating insights from the Nash product bid calculation, the agent aims to initiate negotiations from a position that maximizes the likelihood of reaching satisfactory agreements.
3. Dynamic Threshold Adjustment with Exponential Decay:
    - The acceptance strategy employs exponential decay to dynamically adjust the threshold over time. This adjustment considers factors such as the current negotiation round and the total number of rounds, allowing the agent to adapt to changing negotiation dynamics and time pressure effectively. By gradually decreasing the threshold over time, the agent can navigate negotiations while balancing the need to reach agreements promptly with the desire to maximize utility.
4. Consideration of Offer and Previous Counter Bid Utility:
    - In addition to the threshold, the acceptance strategy now considers the utility of the current offer and our agent's previous bid. If the utility of the current offer exceeds our agent's previous bid, the offer is accepted. This refinement reflects a willingness to accept improved terms and enhances the agent's flexibility in negotiating mutually beneficial agreements.

In summary, all these adjustments aim to optimize the agent's decision-making process, promote efficient negotiation, and maximize utility within the allotted time frame. By incorporating dynamic threshold adjustment and considering the Nash value, our acceptance strategy enhances adaptability, control, and effectiveness in navigating diverse negotiation environments. Through these refinements, our negotiating agent can reach satisfactory agreements while mitigating the risk of unfavorable outcomes, ultimately maximizing utility for itself.

2. Bidding Strategy

Our agent's first proposal seeks to present the finest deal in our best interests by taking into account that the opponent's utility is not below our agent's reservation value. The reason for this is that since we have no information about the opponent's reservation value at the beginning, we try to offer the best for us, without reducing the opponent's utility excessively. Early on, within a specified time frame, our agent will typically follow less concessive behavior by making a little concession to its utility value since our further bid offering strategy highly depends on opponent behavior, we chose this first approach at the beginning of the negotiation. Furthermore, our agent sometimes makes a random offer. The main purpose of this is to confuse the opponent while they are trying to model our agent's behavior.

Our agent's bidding strategy is mainly behavioral. It considers the opponent's concession rate and checks whether the opponent's concession rate is fast or not. After a specified time frame from the beginning and if it doesn't make some random offers, it starts to behave more according to the opponent's concession rate. Moreover, if the utility of the calculated bid is less than the utility of the Nash bid, the agent makes a Nash offer. How many times a Nash offer can happen is counted and kept track. Nash point is the point that maximizes both agent's utility. In other words, it is an equilibrium point that gives them the highest utility.

To understand the logic of our agent's bidding strategy, let's look at the pseudocode of the bidding strategy function.

Pseudocode of bidding strategy:

- *If there is no rational outcome, return None.*
- *If it is the first bid, the agent sends a bid with the highest utility from the rational outcomes (num_bid==0).*
- *If the number of bids is less than and equal to the 5 (specified windowed frame) (num_bid <= 5), our agent's concession rate is adjusted to 0.02. Then, the utility of the current bid is calculated by reducing it with this concession rate and the agent finds the bid from the rational outcomes that has the closest utility to this reduced utility. Return the bid.*
- *If the number of bids is greater than 5 (num_bids > 5), our agent sometimes makes a random offer. It depends on the probability. The probability of making a random offer is 0.1. This random offer is selected from the rational outcomes that have a utility greater than the agent's threshold. Threshold information is kept in the agent's implementation and updated in the acceptance strategy.*
- *If our agent does not make some random offers, it adjusts its concession rate based on the opponent's concession rate. If the opponent concedes fast, our agent determines its rate is the same as the opponent's. Otherwise, the agent adjusts its rate to 80% of the opponent's concession rate.*
- *The agent calculates the utility of its bid after determining its concession rate and seeks the bid from the rational outcomes that have the closest utility to this reduced utility.*
- *If the agent's utility calculation is less than the utility of the Nash bid, and if our agent can still make a Nash offer, the Nash offer is made by the agent.*
- *If the opponent's concession rate is less than 0.005, our agent increases its concession rate by 0.001. Then it calculates the bid and returns it.*

The bidding strategy of our agent mainly depends on the behavior of the opponent's agent, the agent should keep track of the opponent agent's concessions. To realize this, some functions related to this issue are implemented. Let's look at their pseudocode.

Pseudocode of opponent_concession_rate:

- *A specific window size is determined. The window size is adjusted to 5. This is the number of recent bids of the opponent to take into account while calculating the average concession rate of the opponent.*
- *The average concession rate is initialized with 0. In the first 5 bids, it will be 0.*

- *After the first 5 bids, the method always calculates the average of the last 5 bids sent by the opponent and returns the concession rate.*

Pseudocode of opponent_is_conceding_fast:

- *It takes our concession rate as a parameter and checks if the opponent's concession rate is greater than our concession rate. (The opponent's concession rate is calculated with the opponent_concession_rate method.)*

These methods are used to analyze the opponent's behavior in terms of how quickly they are conceding in the negotiation process. The information coming from these functions is used to set our bidding strategy.

3. Opponent Model

The TAK Agent models the opponent's reservation value using a Bayesian technique, which allows it to modify its negotiating strategy simultaneously. The approach taken by the TAK Agent for modeling the opponent's reservation value is covered in this section.
Throughout the negotiating process, the TAK Agent keeps updating its estimation of the opponent's reservation value through the offers put forward by the opponent. To update its estimate of the opponent's reservation value, the agent uses Bayesian estimation. For simplicity, it uses a Gaussian prior distribution, with the agent's reservation value as the prior mean. Using the opponent utilities from the past, the standard deviation of the prior distribution is calculated.

Later on, The agent calculates the posterior distribution of the opponent's reservation value using Bayes' theorem. Taking into consideration both past knowledge and observed data, this distribution represents the most recent estimate of the opponent's true reservation value. Our Bayesian Theorem states that:

$$P(Reservation\ Value|\ Offer) = \frac{P(Offer|Reservation\ Value)\ x\ P(Reservation\ Value)}{P(Offer)} \tag{1}$$

We can explain this theorem with the logic we use in our agent as follows:

1. **Posterior Distribution** $P(Reservation\ Value|Offer)$ **:** Given the observed offer, this is an updated estimate of the opponent's reservation value. In our agent, our estimation corresponds to "prior_mean" which is an updated estimation of the opponent's reservation value.
2. **Likelihood** $P(Offer\ |\ Reservation\ Value)$ **:** This is the likelihood that the offer will be observed considering the reservation value of the opponent. In our agent, we use the known standard deviation "likelihood_std" to compute the likelihood as a Gaussian distribution. We assume that the observed offer originates from a distribution that is centered on the reservation value of the opponent.
3. **Prior Distribution** $P(Reservation\ Value)$**:** This reflects the beliefs we had formed before receiving the offers, on the opponent's reservation value. We assume that the opponent's reservation value may be comparable to our agent's, so we set the prior mean to the agent's own reservation value.

4. **Marginal Likelihood** $P(Offer)$**:** This is the marginal likelihood of observing the offer**.** Since this term acts as a normalization constant to guarantee that the posterior distribution integrates into one, we don't directly compute it in our agent. The estimating procedure is unaffected by that since it is a constant.

The equation from our code for updating the posterior mean (opponent's estimated reservation value) is given below :

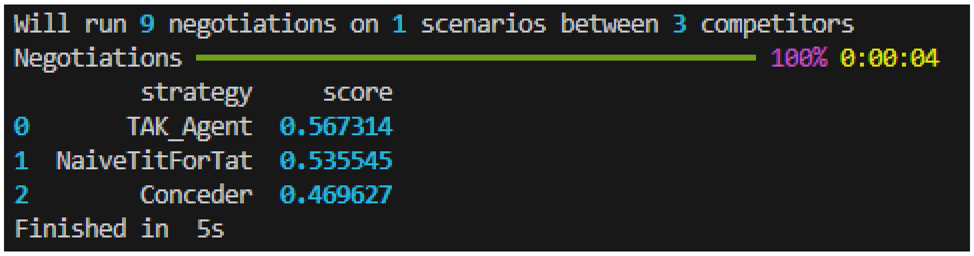
$$\mu' = \frac{posterior\ precision}{prior\ precision} \left(\frac{\mu}{\sigma^2} + \frac{u}{\sigma^2}\right) \tag{2}$$

Where μ is the prior mean, σ is the prior variance, σ' is the likelihood variance, and u is the opponent's utility for that offer. Also prior precision equals $1/\sigma^2$ and posterior precision equals $1/\sigma'^2 + 1/$(prior precision).

Lastly, the agent removes the offers from its collection of rational outcomes that are less than the opponent's estimated reservation value. This guarantees that the agent concentrates on making offers that the opponent will accept.

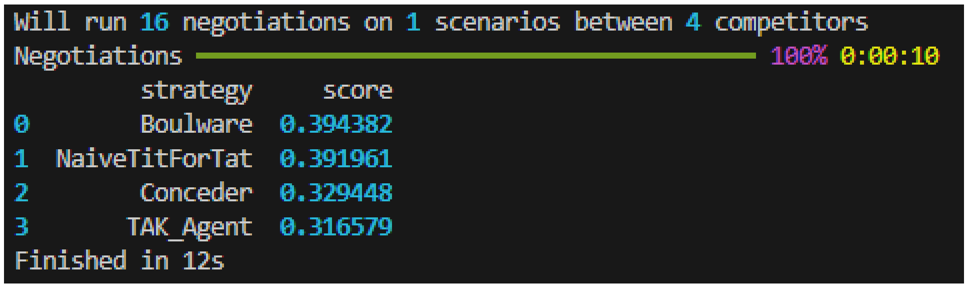
IV. Quantifying the agent's performance

In our testing sessions, we observed a diverse range of outcomes when pitting our agent against itself and default ANL competitor agents in the party domain. Across multiple negotiation sessions with varying preference profiles, our agent demonstrated both impressive successes and notable setbacks. In some instances, our agent ranked first among the negotiators, showcasing its ability to effectively strategize and secure favorable deals. Here is a successful example:



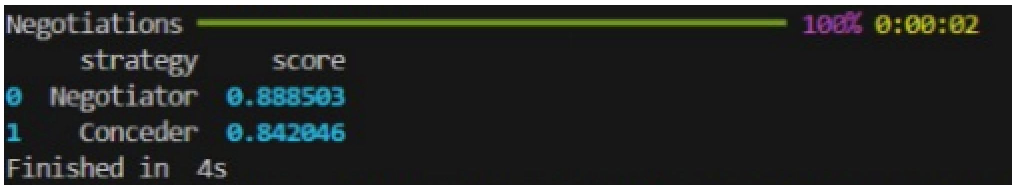
```
Will run 9 negotiations on 1 scenarios between 3 competitors
Negotiations ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:04
         strategy     score
0       TAK_Agent  0.567314
1  NaiveTitForTat  0.535545
2        Conceder  0.469627
Finished in  5s
```

Conversely, there were scenarios where our agent found itself at the bottom of the rankings, indicating areas where its negotiation approach may need refinement. Here is an unsuccessful example:



```
Will run 16 negotiations on 1 scenarios between 4 competitors
Negotiations ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:10
         strategy     score
0        Boulware  0.394382
1  NaiveTitForTat  0.391961
2        Conceder  0.329448
3       TAK_Agent  0.316579
Finished in 12s
```

Upon closer analysis, the outcomes appeared to be influenced by several factors, including the adaptability of our agent's strategies to different opponent behaviors, the complexity of the negotiation environment, and the presence of strategic opportunities. While some outcomes exhibited characteristics of Nash solutions such as:

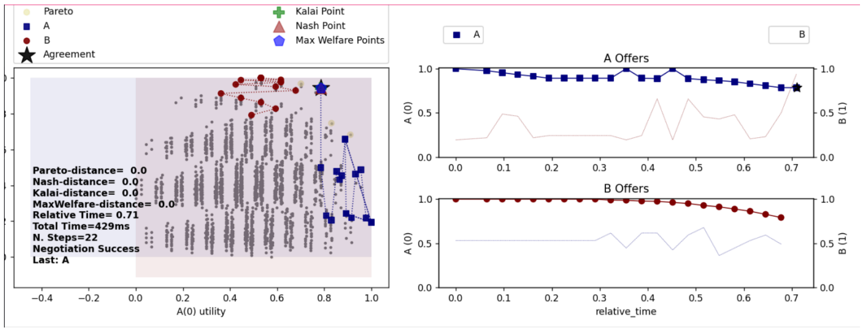


```
Negotiations ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:02
      strategy     score
0  Negotiator  0.888503
1    Conceder  0.842046
Finished in  4s
```

Others hinted at the possibility of reaching efficient outcomes lying on the Pareto Frontier. Achieving such outcomes proved challenging due to the dynamic nature of the negotiation process and the need for effective coordination between parties. Overall, these findings underscore the complexity of negotiating in the party domain and highlight the importance of continued refinement and adaptation of our agent's negotiation strategies.

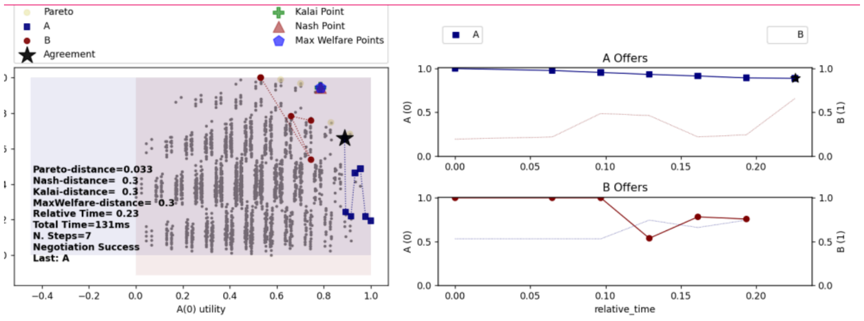
(a) Basic Tests on Party Domain

After doing basic tests on the Party domain, the outcomes look promising. There are agreements on Nash Point, Welfare, and Pareto Frontier which means that it is possible to reach an efficient outcome.



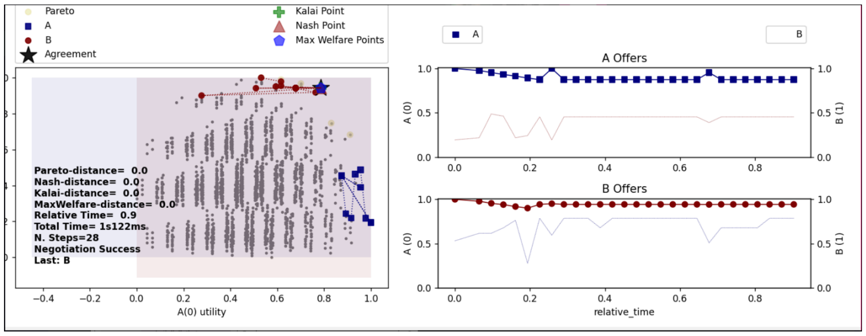
TAK_Agent (A) vs Boulware (B) on Party Domain (Agreement on Nash Point and Welfare)

The agents came to a mutually advantageous understanding of the Nash Point during their negotiation sessions on the Party domain with Boulware and TAK_Agent. This result implies that both agents maximized their own utility while taking the opponent's preferences into account while designing their strategies. Also, both parties reached a welfare-maximizing agreement as a result of the negotiations.



TAK_Agent (A) vs NaiveTitForTat (B) on Party Domain (Agreement on Pareto)

A Pareto-efficient outcome is achieved between TAK_Agent and NaiveTitForTat agent which implies that it's impossible to make any party better off without making another party worse off. In this instance, both agents were successful in navigating the solution space and determining a point at which making any more changes would put one of the parties to the negotiation at risk. Also, our agent's initial greedy approach caused less concession from our side.

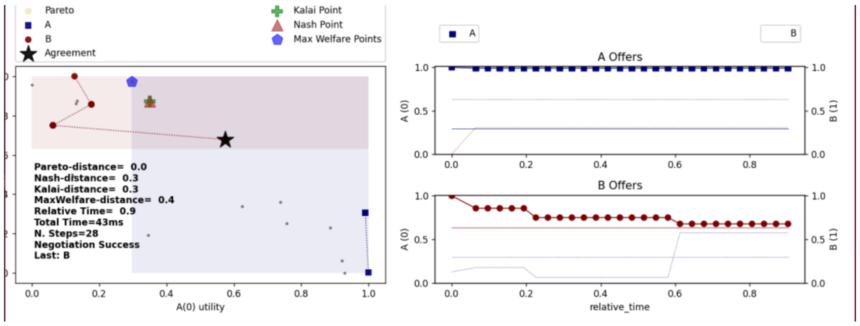


TAK_Agent vs TAK_Agent on Party Domain (Agreement on Nash Point and Welfare)

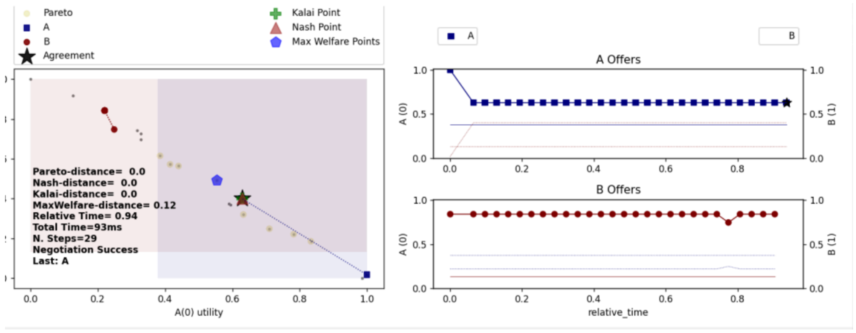
The agents came to a mutually advantageous understanding on the Nash Point during their negotiation sessions on the Party domain against themselves. This result implies that both agents maximized their own utility while taking the opponent's preferences into account while designing their strategies. Also, both parties reached a welfare-maximizing agreement as a result of the negotiations.

(b) Tests on Other Domains

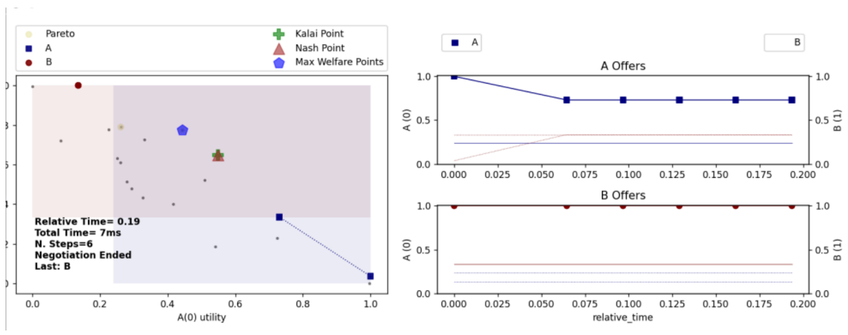
Our agent is generic. It also works on the other domains as well on the party domain. In the tests on other domains (Arbitrary Pie Domain in our case), our agent negotiates against itself, Conceder and Boulware. Conceder and Boulware agents are already provided to us from the standard ANL tournament settings. The outcomes and reports can be seen below:



TAK_Agent (A) vs. Conceder (B) in Arbitrary Pie Domain (Agreement)



TAK_Agent (A) vs. TAK_Agent (B) in Arbitrary Pie Domain (Agreement on Nash Point)
Since our agent was a little persistent in trying to reach an agreement at Nash Point, in this case, the agreement was reached at Nash Point.

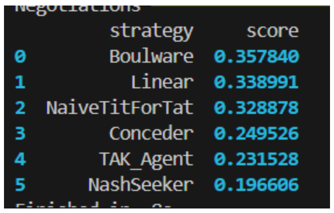


TAK_Agent (A) vs. Boulware (B) in Arbitrary Pie Domain (No agreement).

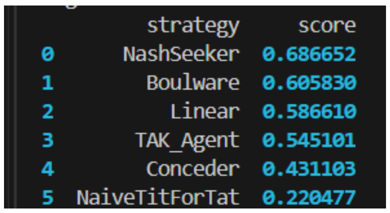
(c) Opponent Model Tests

We experimented both with and without opponent modeling when evaluating our negotiating agent. The goal was to evaluate how opponent modeling affected the agent's performance in various negotiation situations.

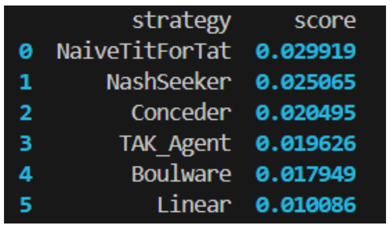
With Opponent Modeling:
We found that when our agent used opponent modeling, it performed very similarly to its competitors, which included the Boulware, Conceder, NaiveTitForTat, Linear, and NashSeeker agents.
Our agent is typically placed between the fourth and fifth spots among these rivals. Although there were minor differences in each round, our agent's total score was similar to the competitors. By using opponent modeling, our agent was able to adjust its tactics more successfully, which produced competitive results.



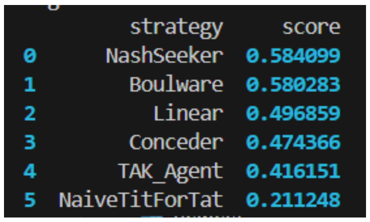
```
         strategy     score
0        Boulware  0.357840
1          Linear  0.338991
2  NaiveTitForTat  0.328878
3        Conceder  0.249526
4       TAK_Agent  0.231528
5      NashSeeker  0.196606
```



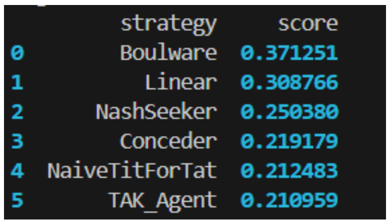
```
         strategy     score
0      NashSeeker  0.686652
1        Boulware  0.605830
2          Linear  0.586610
3       TAK_Agent  0.545101
4        Conceder  0.431103
5  NaiveTitForTat  0.220477
```



```
         strategy     score
0  NaiveTitForTat  0.029919
1      NashSeeker  0.025065
2        Conceder  0.020495
3       TAK_Agent  0.019626
4        Boulware  0.017949
5          Linear  0.010086
```



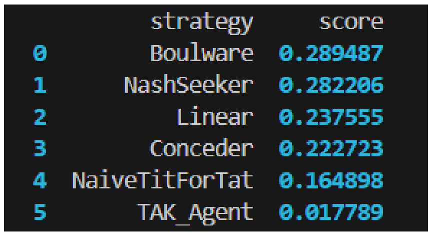
```
         strategy     score
0      NashSeeker  0.584099
1        Boulware  0.580283
2          Linear  0.496859
3        Conceder  0.474366
4       TAK_Agent  0.416151
5  NaiveTitForTat  0.211248
```



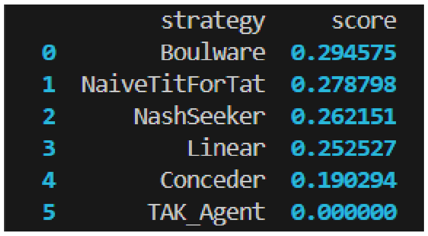
```
         strategy     score
0        Boulware  0.371251
1          Linear  0.308766
2      NashSeeker  0.250380
3        Conceder  0.219179
4  NaiveTitForTat  0.212483
5       TAK_Agent  0.210959
```

Without Opponent Modeling:
On the other hand, our agent performed noticeably worse than the other agents when it didn't have opponent modeling. Among the competitor agents, our agent always came in last or 5th place during the negotiations. Without knowledge of the opponent's behaviors and tendencies, our agent found it difficult to

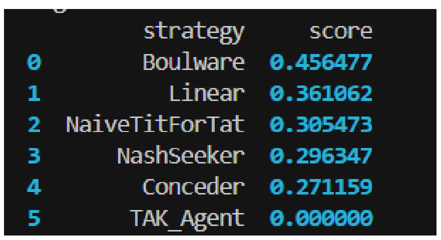
maximize its tactics and produce desired results. As a result, its utility was significantly lower compared to scenarios where opponent modeling was employed.



```
          strategy     score
0         Boulware  0.289487
1       NashSeeker  0.282206
2           Linear  0.237555
3         Conceder  0.222723
4  NaiveTitForTat  0.164898
5        TAK_Agent  0.017789
```



```
          strategy     score
0         Boulware  0.294575
1  NaiveTitForTat  0.278798
2       NashSeeker  0.262151
3           Linear  0.252527
4         Conceder  0.190294
5        TAK_Agent  0.000000
```



```
          strategy     score
0         Boulware  0.456477
1           Linear  0.361062
2  NaiveTitForTat  0.305473
3       NashSeeker  0.296347
4         Conceder  0.271159
5        TAK_Agent  0.000000
```

V. Future perspectives

To make negotiation agents more useful in real-life situations, several improvements are needed. First, they should be able to make smarter decisions by considering multiple factors at once. Second, they need to understand human language better, including emotions, for more effective communication. Third, agents should be equipped to recognize and respond to human emotions to make negotiations smoother. Fourth, users should have the ability to customize negotiation strategies based on their goals. Fifth, providing real-time feedback during negotiations can help users make better decisions. Sixth, enabling agents to communicate through different channels like voice and gestures would make interactions more natural. Finally, ensuring ethical and legal compliance in negotiations is crucial for maintaining trust and avoiding issues. These enhancements would make negotiation agents more practical and capable of achieving better deals in real-life scenarios.

VI. Conclusion

From the beginning to the end, we worked as a team helping each other when needed and doing our tasks on time. At first, since we had only theoretical knowledge about negotiation agents, the environment and components seemed very different and kind of hard to understand. Since we had no experience in the environment, it was even more challenging to write a design report without knowing if our solutions would be feasible. After completing the design and moving on to the implementation, we learned our limits, boundaries, and the things we thought were good but were not in the negotiation environment. We modified our approaches accordingly and finalized our agent implementation as a team. As I mentioned, we helped each other up whenever we needed but to be more specific about the task distribution:
Kerem was responsible for the negotiation domain analysis in the first report and for acceptance strategy implementation and writing its part in the final report. Also, he was responsible for the Introduction, the beginning of part 4 and part 4.a, and the conclusion in the final report. He also touched upon and improved different parts of our project implementation.
Alkın was responsible for the Design of the Negotiation Strategy with Tuğçe in the first report and for bidding strategy implementation and writing the bidding strategy part in the final report. He has also contributed to the implementation of acceptance strategy and opponent model sections. Moreover, he was also responsible for the High-Level Description of the Agent part and for part 4.b.
Tuğçe was responsible for the Design of the Negotiation Strategy with Alkın in the first report and for the opponent modeling implementation and writing its part in the final report. Also, she was responsible for part 4.c in the final report and also helped with parts 4.a and 4.b, other than that, she touched upon and improved different parts of our project implementation.