UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

33977 - Negociación Automática y Resolución de Conflictos

Negotiation With BOA and Automatic Learning

Author:
Musæus, Lars Gjardar

26th May 2021

Contents

1 Introduction

This project had two parts, where the first part was to pick out some opponents, three domains and optimize a BOA agent against these four opponents. The second part was to design a *Opponent Model* BOA component.

2 Part 1 - Optimization Of Bilateral Agents With BOA

In this first part we are going to use already implemented BOA components to optimize a agent for bilateral negociation against four opponents. To make the agent as good as possible we have to carefully select the *domains* and *opponents* such that the agent is as robust as possible.

2.1 Requirements

In this part we have a certain set of requirements which are the following:

- Bilateral negotiations.
- Without learning between negotiation sessions.
- 1000 discrete rounds of negotiation.
- Selection of three domains of different size, that are not discounted.
- Optimized against 4 appropirate agents.

2.2 Domain

We start off by putting our focus on the selection of our three domains. As we want our agent to be as good as possible, there are a few things one need to have in mind. The two measurable factors i will look at is the domains' *size* and *level of opposition*.

2.2.0.1 Size

The size of the domain is an important factor for the quality of a opponent model, and most of the time a bigger domain will reduce the likelyhood of a Pareto-bid. It's also important to mention that the bigger the domain is the more computation power/time is needed, obviously. Thus this affects the time/exploration trade off. [2]

2.2.0.2 Opposition

By opposition we mean in what degree the parties in a domain are able to come to come to an agreement. In other words, how many possible agreements exists.

As both *size* and *opposition* influences the opponent model, it makes sense that the three domains have a wide spread in these factors. Based on this I ended up with the domains in' Table 1. Under the *Opposition* column have written *high*, *med* and *low* based on [10].

Table 1: The three chosen Domains

Domain	Size	Opposition
Itex vs Cypress	180	0.43146(high)
England Zimbabwe	432	0.27212(med)
Party	7200	0.20880(low)

2.3 Opponents

Now that the domains are chosen it is time to choose the 4 opponents. As beforementioned the goal for our agent is to be the best possible, thus my strategy for choosing opponents is to pick the best competitors i.e. the winners of the Automated Negotiation Agent Contest(ANAC) the last 4 competitions that are implemented and available in Genius. The following four agents are the winners of the category of having highest individual utility in 2018, 2017, 2016 and 2015.

2.3.0.1 AgreeableAgent - ANAC 2018

Our first opponent is the winner of ANAC 2018[8], AgreeableAgent. With a time-based strategy the AgreeableAgent uses the history of bids in a given session to learns its opponent model[9].

2.3.0.2 PonPokoAgent - ANAC 2017

The second opponent is the winner of ANAC 2017[7]. With several different bidding strateges, PonPokoAgent randomly selects a strategy from its reportuar for each session. By doing this its strategy will seem random and is very hard for the opponent agent to predict its strategy[1].

2.3.0.3 Caduceus - ANAC 2016

The winner of ANAC 2016[6], Caduceus eploys a strategy called *incremental portfolio strategy*. The strategy puts weights on experts, and then asks the most reliable expert what to do each negotiation round[4].

2.3.0.4 Atlas3 - ANAC 2015

Finally we have Atlas3, the winner of ANAC 2015[5]. Atlas3 employs relative utility search for linear utility spaces, where relative utility is based on maximum utility. It also uses the frequency of the opponent's bidding history to apply a replacement method[3].

2.4 BOA Components

As it exists a ton of combinations, it will take a man age to try every possible comination of components, and thus the picking has to be a bit random. I have decided to to pick 3 *Opponent Models*, 3 *Bidding Strategies*, 2 *Acceptance Strategies* and use the *Best Bid Opponent Model Strategy* in every combination, because Best Bid always tries to give the best possible bid. This gives us 18 different combinations/agents that we will run in tournements against our four opponents, and by doing a gridsearch we will end up with the best agent. Table 2 shows the different combinations. There is not a column for *Opponent Model Strategy* because all agents use the same i.e. *BestBid*.

2.5 Testing

To find out which BOA agent is the best we start off by running all the agents in three different tournaments, with 1000 rounds each. Genius returns all the stats from the tournament in a .csv file which we can do statistical research on. What we want to do is to see if the utility of a certain agent is significantly different - better - than the other agents. To do this we can use Kruskal Wallis and post hoc dunn's test. Kruskal Wallis goes through the utilities of the different domains and sees if something is statistically significantly different, but it does not say what and where. That is the job for the post hoc Dunn's test. By looking at the result from the post hoc Dunn's

test, as well by looking at the average utilities of the different BOA agents we hopefully can draw a conclusion.

Table 2: Combination of BOA Components

Agent Name	**Bidding Strategy**	**Acceptance Strategy**	**Opponent Model**
boa_0	2012 - AgentLG	2011 - TheNegotiator	Perfect Model
boa_1	2012 - AgentLG	2011 - TheNegotiator	Bayesian Model
boa_2	2012 - AgentLG	2011 - TheNegotiator	Agent X Frequency Model
boa_3	2012 - AgentLG	2011 - HardHeaded	Perfect Model
boa_4	2012 - AgentLG	2011 - HardHeaded	Bayesian Model
boa_5	2012 - AgentLG	2011 - HardHeaded	Agent X Frequency Model
boa_6	2012 - IAMHaggler	2011 - TheNegotiator	Perfect Model
boa_7	2012 - IAMHaggler	2011 - TheNegotiator	Bayesian Model
boa_8	2012 - IAMHaggler	2011 - TheNegotiator	Agent X Frequency Model
boa_9	2012 - IAMHaggler	2011 - HardHeaded	Perfect Model
boa_10	2012 - IAMHaggler	2011 - HardHeaded	Bayesian Model
boa_11	2012 - IAMHaggler	2011 - HardHeaded	Agent X Frequency Model
boa_12	2011 - AgentK2	2011 - TheNegotiator	Perfect Model
boa_13	2011 - AgentK2	2011 - TheNegotiator	Bayesian Model
boa_14	2011 - AgentK2	2011 - TheNegotiator	Agent X Frequency Model
boa_15	2011 - AgentK2	2011 - HardHeaded	Perfect Model
boa_16	2011 - AgentK2	2011 - HardHeaded	Bayesian Model
boa_17	2011 - AgentK2	2011 - HardHeaded	Agent X Frequency Model

3 Part 2 - Programming of BOA Component

In the second part of the delivery we are supposed to program a BOA component, more specifically a *Opponent Model*.

3.1 The Model

The model I wrote uses the concepts from the *frequency model* by looking at the frequency of certain events it will set the weights of the opponents utility based on this. The frequency model checks if attributes has the same values as the previous received offers, which could imply that the opponent has a higher utility with these certain values. Thus the weights for these values are increased. My model does the same, but insteaf of just using the estimated utility calculated by the frequency model, it compares this to the estimated opponent utility calculated by the original *getUtility()* function. This function also sets weights on the different issues, and uses this to estimate the utility. As the frequency model has estimated a utility, the same with the original model they compare to eachother, if the difference is small it's likely that the estimation is close to the actual utility, thus the returned utility will be in between the two models.

Bibliography

[1] Reyhan Aydoğan. *ANAC 2017: Repeated Multilateral Negotiation League*. URL: https://link.springer.com/chapter/10.1007/978-981-15-5869-6_7 (visited on 29th Apr. 2021).

[2] Tim Baarslag. *Measuring the Performance of Online Opponent Models in Automated Bilateral Negotiation*. URL: https://www.researchgate.net/publication/257232252_Measuring_the_Performance_of_Online_Opponent_Models_in_Automated_Bilateral_Negotiation (visited on 29th Apr. 2021).

[3] Katsuhide Fujita. *The Sixth Automated Negotiating Agents Competition (ANAC 2015)*. URL: https://link.springer.com/chapter/10.1007/978-3-319-51563-2_9 (visited on 29th Apr. 2021).

[4] Taha D. Güneş. *Collective Voice of Experts in Multilateral Negotiation*. URL: https://link.springer.com/chapter/10.1007/978-3-319-69131-2_27 (visited on 29th Apr. 2021).

[5] *Official website for ANAC 2015*. URL: http://web.tuat.ac.jp/~katfuji/ANAC2015/ (visited on 29th Apr. 2021).

[6] *Official website for ANAC 2016*. URL: http://web.tuat.ac.jp/~katfuji/ANAC2016/ (visited on 29th Apr. 2021).

[7] *Official website for ANAC 2017*. URL: http://web.tuat.ac.jp/~katfuji/ANAC2017/ (visited on 29th Apr. 2021).

[8] *Official website for ANAC 2018*. URL: http://web.tuat.ac.jp/~katfuji/ANAC2018/ (visited on 29th Apr. 2021).

[9] Y. Ohsawa. *Advances in Artificial Intelligence*. URL: https://www.springer.com/gp/book/9783030398774 (visited on 29th Apr. 2021).

[10] Colin R. Williams. *An Overview of the Results and Insights from the Third Automated Negotiating Agents Competitionn*. URL: https://link.springer.com/chapter/10.1007/978-4-431-54758-7_9 (visited on 29th Apr. 2021).

Appendix

A Link to project on GitHub

A.1 Hyperlink

Project on GitHub

A.2 Text

https://github.com/larsgmu/NAC/tree/main/Project2