Problem Solving by Negotiation: a practical assignment for IN4010 Artificial Intelligence Techniques

Catholijn M. Jonker and Pradeep Murukannaiah

Academic year 2020-2021

1 Introduction

The practical assignments of the IN4010 course on Artificial Intelligence Techniques allow you to get first hand experience with some AI techniques. To better understand their applicability, to translate suitable problems to fit the scheme of the technique, and how to interpret their results.

In this practical assignment you will learn to use the power of automated multi-party negotiation. Negotiation is a technique that can be used in conditions of (partially) incomplete and uncertain information with many stakeholders. This assignment gives you the opportunity to taste the power of distributed artificial intelligence to solve well-known complex problems through a multi-agent approach in the form of multi-party negotiation.

1.1 Learning Objectives

The learning objectives of this assignment are

- To learn and apply techniques and theories from multi-agent negotiation, such as preference modeling and utility theory, group decision-making, communication protocols and coordination in multi-agent systems, opponent modeling, and strategy modeling.
- To learn to translate group decision problems into automated negotiation problems.
- To learn to design and analyze a negotiating agent for a realistic domain, including among others a negotiation strategy.
- To actively discuss in student groups the translation of a group decision making problem into a multi-agent negotiation problem and to jointly coordinate the design and construction of the domain of negotiation, the party profiles, and a negotiating agent.

In this document you will find the outline of the assignment and pointers to the literature that you can use as background information. Furthermore, you will find the links to the negotiation platform that we will use for this assignment. We also provide the usual information on deadlines, where to get help, how to submit your files, and so on. For any remaining questions we refer you to Brightspace.

This document is organized as follows. The core of the assignment is provided in 2. Organizational details of deadlines and what to hand in and so on, can be found in 3. To help you get started, 5 provides a list of questions and steps to familiarize you with the negotiation environment. References to more information on negotiation and group decision problems are available in 6.

2 The assignment

2.1 List of Problems you can choose from

To experience the strength of multi-party negotiation as a form of distributed artificial intelligence, we ask you to pick one of the problems listed below and translate that to a negotiation problem.

- Meeting scheduling, for inspiration on another distributed AI approach, see e.g., [21]
- A form of the coalition formation problem, see [6] for definitions. For example, you can choose
  - A hedonic game
  - Popular partitioning
  - Government Coalition Formation
  - Buyer coalition formation (for inspiration, see e.g., [25], [17])

2.2 Assignment in Steps

The assignment consists of the following parts and includes the writing of a report on all parts. Section 3.1 details what you have to deliver, in what format and when.

1. Pick a problem from the list
2. Explain how the problem fits the MOPAC negotiation protocol
3. Translate the problem to a MOPAC domain model with appropriate preference profiles (linear additive), and party descriptions (name, power).
4. Develop an automated negotiating agent that can run in any negotiation under the MOPAC protocol for any domain.
5. Run the negotiations repeatedly (or run tournaments), see Section 2.3.
6. Discuss to what extent the outcome of the negotiations solves the problem.
7. Indicate whether you are interested in joining a scientific team to write a paper about the assignment, whether or not you want to participate in the competition and/or whether you are interested in doing a master thesis/research on this topic. See Section 7 for more information.

2.3 Analyzing the Performance of Your Party

How well does your party perform and how generic it? That is, does it perform well against other parties? And does it work as well on the domain you initially developed the party for as on other domains? To verify this, have your party negotiate against itself on the domain you created, and against at as many other parties you can find. For this you can collaborate with other groups to ask them for their party, or look in the GENIUSWEB repository for parties that can run with this protocol. If you collaborate with other groups, explicitly name and thank these groups in your report. Report on the outcomes reached. Try to explain why the outcome is as it is.

2.4 Ethics and Plagiarism

We expect from our students that they adhere to the ethical standards of TU Delft, which includes that the work you submit is that of your group only, and you only collaborate with other groups or people in as far as is explicitly allowed by the assignment. In particular, the following behaviors are strictly prohibited:

- Designing an party in such a way that it benefits some specific other party.
- Hacking or exploiting bugs in the software.

- Communicating with the party during time it is being evaluated by the teaching team.

Both the agent source code and the report need to be your own work. For the party, you can use the code from example agents provided, but anything else needs to be clearly acknowledged in the report and when submitting your work. Note that you are not allowed to directly use code programmed by others, including parties who have previously participated in the international competition. However, you are allowed to use ideas and strategies reported in academic papers, as long as you implement these strategies yourselves and you acknowledge the papers in your report. In case of doubt, feel free to ask! This is important as violations, deliberate or otherwise, will be reported.

2.5 Open Science

Finally, please do realise that your party and domain model, preferences profiles and party descriptions will be open source and will be shared with the world. We expect of our students that they come up with non-offensive material according to TU Delft standards.

3 Deliverables, Requirements, and Submission Details

This section details how the practical work is organized and where, when and what you have to deliver. In short, you should deliver a zip file containing your report, code of your main class (and possibly helper classes), the domain and preference profile files, and a jar file of your party.

3.1 Deliverables

- Your group number n should be clearly stated in every document and every file you submit. The assigned group number is the number of your group on Brightspace. Your end report should list the names of the group members along with their student numbers.
- A negotiating party programmed in Java or Python (note that regarding Python, most support is offered for Jython) using GENIUSWEB. Your submission should consist of a jar file of your party and a package containing your source code: both the class and src files. It is obligatory to use the package "ai2020.group" + n for the negotiating party and any other required files. A party consists of a main class and optionally several helper classes. For example for group 3 the package could include: *Group3_Main*, *Group3_Helper1*, and *Group3_Helper2*.
- A folder containing the JSON files for the domain and preference profiles (should be linear additive profiles) that you created to demonstrate how negotiating parties can solve the problem you chose.
- A report discussing your choices and findings regarding each part of the assignment. In particular, we point out to you that we need:
  - A high-level description of the problem you picked from the list with an explanation of how the problem fits the MOPAC negotiation protocol and a description of the "translation" to a MOPAC domain model with appropriate preference profiles, and party descriptions (name, power).
  - A high-level description of the party and its structure, including the main Java (or Python) methods (mention these explicitly!) used in the negotiating party that have been implemented in the source code,
  - An explanation of the negotiation strategy, decision function for accepting offers, any important preparatory steps, and heuristics that the party uses to decide what to do next,
  - A section documenting the strong and weak points of your party, the tests you performed to analyze (and improve) the negotiation strength of your party. You must include scores of various tests over multiple sessions that you performed. Describe how you set up the testing situation and how you used the results to modify your party, and motivate why you choose these testing methods/metrics/graphs. See Section 5.1 for more information.

- Cover all the points of Section 2.2.

The report should be concise (5 A4 pages maximum), but should include an explanation and motivation of *all* of the choices made in the design of negotiating party. The report should also help the reader to understand the organization of the source code (important details should be commented on in the source code itself). This means that the main Java (or Python) methods used by your agent should be explained in the report itself.

Please make sure all your files are in the directory structure as explained above. Assuming that the group has number three, a valid directory structure is:

```
Group3_report.pdf
ai2020/
    group3/
        Group3_Main.class
        Group3_Main.java
        HelperClass1.class
        HelperClass1.java
        HelperClass2.class
        HelperClass2.java
    group3_yourDomain/
        group3_yourDomain.json
        group3_preferenceProfile1.json
        group3_preferenceProfile2.json
        group3_preferenceProfile3.json
```

3.2 Requirements

- The party should implement a negotiation strategy to generate offers, and an acceptance strategy to decide whether to accept an offer or not.
- The party must aim at the best negotiation outcome possible, while taking into account that it may need to concede to the other parties.
- Show how your party takes the preferences of the other parties into account.
- The party meets reasonable time and memory constraints. Any party that uses more than a minute in order to initiate the negotiation or to reply to an opponent's offer will be disqualified.
- Make sure your party at least works with linear additive profiles with discrete issues.
- The source code should contain explanatory comments that will allow people not involved in programming the code to understand it. A clear explanation of a method must be provided at the beginning of important methods. Parts of the code that are not self-explaining should receive additional documentation. Please incorporate enough information in comments in the code to ensure this. Finally, make sure that you have removed all debug printouts from your code and that your party does not unnecessarily burden the servers.
- Your code should be tested to ensure that it works on multiple operating systems, and requires nothing other than the files contained in your group's package. Note: please make sure your submitted files are self-contained and can be added to GeniusWeb in the standard way: that is, please zip your entire maven project so that we can build your party by unzipping it and then running "mvn package".
- The report should include all the elements listed in Section 3.1.

3.3 Submission

Assignment Deadline: **November 5 2020, 17:00**. Deadline for submitting your team solution including a negotiating software agent and report.

Please submit your report in PDF format and the package with all your code on **Brightspace**. Use the naming conventions described elsewhere in this document, including your group number. Do not submit incomplete assignment solutions; only a complete assignment solution containing all deliverables will be accepted. The deadline for submitting the assignment is strict. In case of any problems, please contact us well in advance of the deadline.

4 Evaluation

Assignments are evaluated based on several criteria. All assignments need to be complete and satisfy the requirements stated in the detailed assignment description section above. *Incomplete assignments are not evaluated*. That is, if any of the deliverables is incomplete or missing (or fails to run), then the assignment is not evaluated.

The assignment will be evaluated using the following evaluation criteria:

- *Quality of the deliverables*: Overall explanation and motivation of how you translated the problem of your choice to a negotiation problem. The overall explanation of the design of your negotiating party; Quality and completeness of answers and explanations provided to the questions posed in the assignment; Explanatory comments in source code, quality of documentation.
- *Performance:* How well your translation and the negotiating parties solve the original problem.
- *Originality:* Any original features of the problem translation, the domain & profiles formalization, and the negotiating party are evaluated positively. Note that you are required to submit *original* source code designed and written by your own team. Source code that is very similar to that of known negotiating agents or parties or that of other students will not be evaluated and be judged as insufficient.
- *Ethics & Plagiarism:* Detection of fraud or plagiarism, and any unethical behaviour will be reported to the administration.

5 Familiarizing Yourself With the Negotiation Environment

To prepare for the assignment it helps to familiarize yourself with the GeniusWeb environment. For that purpose we prepared a number of questions for you. You do not have to discuss the questions in 5 in your report. Go to the GeniusWeb website, to find the latest release and read its Information for Students. In order to familiarize yourself with the environment, complete the items below

i Select the option to start a new negotiation session with three negotiating parties that need a consensus deal.

ii Pick an appropriate protocol for the negotiation task

iii Pick a negotiation domain from the available domains

iv Iteratively pick a party from the available parties for this protocol, and pick a profile for that party and then add that party

v When you have the right number of parties, each having the right profile, start the negotiation session.

vi Once the negotiation is over, click on the log file and on the option to render a utilities plot and analyze the performance of the parties.

vii run a couple more sessions in which you let one party play against itself. Is a Pareto optimal outcome reached?

viii Now let it play against another party and run the same negotiation again. Think about why they obtain the outcomes that they do.

ix Now we know how to run a single negotiation session, it is time to run a tournament. In the user guide it is discussed how to run a tournament. Specify a simple tournament with the following parameters:

- NUMBER OF TOURNAMENTS: *1*.
- NUMBER OF TEAMS IN EACH SESSION: *4*.
- PICK TEAMS WITH RETURN: *false*.
- SESSION PROTOCOL : *Multiple Offers Protocol for Multilateral Negotiations with Partial Consensus (MOPaC)*.
- VOTING EVALUATOR: *Largest Agreements and Repeat*.
- DEADLINE : *10 rounds*.
- DOMAIN: *Party*.
- PREFERENCE PROFILES: *Add: Four different profiles in the party domain*.
- TEAMS: *Add Randomparty-x.y.z four times* (other parties don't support MOPaC).

Click on "Start Tournament". Give GENIUSWEB some time to run the tournament. After running your tournament there should be two logs, on the bottom of the page you have a link to the log file and the results table. You might have to refresh to clear the cache, in case you already have an older result. Analyze the logs by importing them with a script or in a spreadsheet and using pivot tables. For Excel users, note that these features are only available in the full version.

x Finally, check the example code available on Trac and Information for Students, and read how to create and analyze your own agent.

5.1 Quantify the performance of your agent

As a final remark, we want to make clear that it is important to test your party in order to improve the negotiation strength of your party in *different* settings. A warning is in place: do not assume that your goal is to outperform a 'stupid' party. Test your party against as many parties as you can and older versions of your own party.

Think carefully on the criteria that you set to judge the performance of your agent. In normal negotiations, a good outcome is determined by criteria, such as the efficiency of the outcome (i.e. is it close or not to the Pareto Frontier), how close is it to the Nash Product, does it optimize Social Welfare, or the utility of the party it represents? However, if you use negotiation to solve other problems, you will have to set the criteria to meet that problem.

There are many techniques to improve the negotiation strength of agents. Various factors can be taken into account when deciding on acceptance, breaking off, or computing a next offer in a negotiation, for example you can take into account the time an opponent takes to reply with a counter offer, the consistency of an opponent's offers, the concession rate of your opponent, and possibly other considerations about the domain of negotiation itself.

6 Background Reading

Some techniques relevant for building and designing negotiating agents can be found among others in chapters 16 and 17 in [20], chapter 2.3 in [23], [19], and [15]. You might also want to go beyond the standard course material. Additional information is provided to you in the form of several papers [2, 13, 18, 8, 22]. There is a lot of other literature available about negotiation that may help you finish this assignment successfully. As for almost any subject, you can find more information about negotiation strategies online. We recommend to search for strategies used in the ANAC competition [1, 5, 7, 9, 10, 14, 16, 24].

7 Future Perspectives

Did you like thinking about efficient negotiating strategies, or implementing a successful negotiating party? You can enter in the international competition, we'd like to help you with that. You could a writing team on the scientific aspects of this assignment, or you could do a master thesis. Negotiation provides a lot of subjects to do your Master Thesis on! You can always ask the course assistants and teachers. You can already find some ideas in this section. We are looking forward to talk to you or receive your email requesting for more information: **pa.ai-ewi@lists.tudelft.nl**.

7.1 International competition

If your agent performs well enough, you could even participate in the yearly competition for automated negotiating agents (ANAC), see e.g., [1, 3, 12]. Student teams that submitted their agent after this assignment have received prizes in the competition (and won $1000,- in 2011 and 2013) and were awarded student scholarships to attend international conferences to present their agent.

7.2 Some ideas for a master project

Negotiation has become a major and interesting research area within Artificial Intelligence. Negotiation is important in many domains ranging from business applications such as Internet market places to incident and crisis management. Negotiation is one of the main tools an autonomous agent has to agree about a course of action or to resolve conflicts with other agents.

The agents in the negotiation environment have limited capabilities in order to achieve a negotiated deal. What extensions would be required to use your agent in real-life negotiations to support (or even take over) negotiations performed by humans? Can you think of additional capabilities (e.g. other actions or forms of communication) for your agent that would make it more practical and help achieve better deals? For interesting challenges, see, e.g., [11, 4]. There are many angles that you can take:

- Design of negotiation strategies and better opponent models (using machine learning techniques);
- Design of a negotiation support system, either advising humans in a particular domain, or even taking over negotiation all together;
- Design of richer negotiation protocols, such as protocols allowing for partial bids, use a form of communication between the agents, auctions;
- Design and add explanation facilities for negotiation support systems, negotiating agents, and/or GENIUSWEB;
- Design and implement agents that can handle non-linear utility functions, add dependencies, and enrich GENIUSWEB to support people using that functionality;
- Design self-learning agents that improve their negotiation strategies with experience;
- Research related to negotiation, trust and the reputation of agents, either cognitively motivated or viewed from a more engineering point of view;
- Research on negotiation and culture. How does culture influence negotiation between different agents?

This can all be combined with a Literature Survey course, in order to kick-start your thesis. If you are interested, don't hesitate to ask us: **pa.ai-ewi@lists.tudelft.nl**

Acknowledgements

We would like to thank dr. W. Pasman for implementing the MOPAC protocol, Rolf Starre for helping out with the writing of the assignment and Jinke He, Daniele Foffano and Nikhil Saldanha for testing.

References

[1] Tim Baarslag, Reyhan Aydoğan, Koen V. Hindriks, Katsuhide Fuijita, Takayuki Ito, and Catholijn M. Jonker. The automated negotiating agents competition, 2010-2015. *AI Magazine*, 36(4):115–118, 12/2015 2015.

[2] Tim Baarslag, Koen V. Hindriks, Mark J.C. Hendrikx, Alex S.Y. Dirkzwager, and Catholijn M. Jonker. Decoupling negotiating agents to explore the space of negotiation strategies. In *Proceedings of The Fifth International Workshop on Agent-based Complex Automated Negotiations (ACAN 2012)*, 2012. URL: https://homepages.cwi.nl/~baarslag/pub/Decoupling_Negotiating_Agents_to_Explore_the_Space_of_Negotiation_Strategies_ACAN_2012.pdf.

[3] Tim Baarslag, Koen V. Hindriks, Catholijn M. Jonker, Sarit Kraus, and Raz Lin. The first automated negotiating agents competition (ANAC 2010). In Takayuki Ito, Minjie Zhang, Valentin Robu, Shaheen Fatima, and Tokuro Matsuo, editors, *New Trends in Agent-based Complex Automated Negotiations*, volume 383 of *Studies in Computational Intelligence*, pages 113–135, Berlin, Heidelberg, 2012. Springer-Verlag. URL: https://homepages.cwi.nl/~baarslag/pub/The_First_Automated_Negotiating_Agents_Competition_ANAC_2010.pdf.

[4] Tim Baarslag, Michael Kaisers, Enrico H. Gerding, Catholijn M. Jonker, and Jonathan Gratch. When will negotiation agents be able to represent us? The challenges and opportunities for autonomous negotiators. In *Proceedings of the Twenty-sixth International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 4684–4690, 2017.

[5] Mai Ben Adar, Nadav Sofy, and Avshalom Elimelech. Gahboninho: Strategy for balancing pressure and compromise in automated negotiation. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 205–208. Springer Berlin Heidelberg, 2013.

[6] Felix Brandt and Martin Bullinger. Finding and recognizing popular coalition structures. In *AAMAS*, pages 195–203, 2020.

[7] A.S.Y. Dirkzwager, M.J.C. Hendrikx, and J.R. Ruiter. The negotiator: A dynamic strategy for bilateral negotiations with time-based discounts. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 217–221. Springer Berlin Heidelberg, 2013.

[8] P. Faratin, C. Sierra, and N.R. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3):159–182, 1998.

[9] Radmila Fishel, Maya Bercovitch, and Ya?akov(Kobi) Gal. Bram agent. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 213–216. Springer Berlin Heidelberg, 2013.

[10] Asaf Frieder and Gal Miller. Value model agent: A novel preference profiler for negotiation with agents. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 199–203. Springer Berlin Heidelberg, 2013.

[11] Nicholas R. Jennings, Peyman Faratin, Alessio R. Lomuscio, Simon Parsons, Michael J. Wooldridge, and Carles Sierra. Automated negotiation: Prospects, methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.

[12] Catholijn M. Jonker, Reyhan Aydoğan, Tim Baarslag, Katsuhide Fujita, Takayuki Ito, and Koen Hindriks. Automated negotiating agents competition (ANAC). In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence and Twenty-Ninth Innovative Applications of Artificial Intelligence Conference*, pages 5070–5072, Feb 2017.

[13] C.M. Jonker and J. Treur. An agent architecture for multi-attribute negotiation. In *International joint conference on artificial intelligence*, volume 17, pages 1195–1201. LAWRENCE ERLBAUM ASSOCIATES LTD, 2001.

[14] Shogo Kawaguchi, Katsuhide Fujita, and Takayuki Ito. Agentk2: Compromising strategy based on estimated maximum utility for automated negotiating agents. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 235–241. Springer Berlin Heidelberg, 2013.

[15] Ralph L. Keeney and Howard Raiffa. *Decisions with Multiple Objectives*. Cambridge University Press, 1976.

[16] Thijs Krimpen, Daphne Looije, and Siamak Hajizadeh. Hardheaded. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 223–227. Springer Berlin Heidelberg, 2013.

[17] Cuihong Li, Katia Sycara, and Alan Scheller-Wolf. Combinatorial coalition formation for multi-item group-buying with heterogeneous customers. *Decision Support Systems*, 49(1):1–13, 2010.

[18] Raz Lin, Sarit Kraus, Tim Baarslag, Dmytro Tykhonov, Koen Hindriks, and Catholijn M. Jonker. Genius: An integrated environment for supporting the design of generic automated negotiators. *Computational Intelligence*, 2012. URL = http://mmi.tudelft.nl/sites/default/files/genius.pdf.

[19] Howard Raiffa. *The art and science of negotiation: How to resolve conflicts and get the best out of bargaining*. Harvard University Press, Cambridge, MA, 1982.

[20] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition, 2010.

[21] Hajer Salem and Ahlem Ben Hassine. Meeting scheduling based on swarm intelligence. *Procedia Computer Science*, 60:1081–1091, 2015.

[22] Roberto Serrano. Bargaining, 2005. URL = http://levine.sscnet.ucla.edu/econ504/bargaining.pdf, November, 2005.

[23] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009.

[24] Colin R. Williams, Valentin Robu, Enrico H. Gerding, and Nicholas R. Jennings. Iamhaggler2011: A gaussian process regression based negotiation agent. In Takayuki Ito, Minjie Zhang, Valentin Robu, and Tokuro Matsuo, editors, *Complex Automated Negotiations: Theories, Models, and Software Competitions*, volume 435 of *Studies in Computational Intelligence*, pages 209–212. Springer Berlin Heidelberg, 2013.

[25] Junichi Yamamoto and Katia Sycara. A stable and efficient buyer coalition formation scheme for e-marketplaces. In *Proceedings of the fifth international conference on Autonomous agents*, pages 576–583, 2001.