CSE 482: Assignment on Automated Negotiation (Prepared in collaboration with Delft University of Technology and Özyeğin University)

Monsoon 2016

1 Introduction

Imagine that you just got a new job in a new city. You need to look for a place to stay and you have various factors that you need to look at. A few of your friends and colleagues are also in a similar situation so you all decide to look for a place together by negotiating with each other and reaching a solution.

The assignment must be completed in teams of 2 students. This assignment aims at making you familiar with *negotiation*. Negotiation is a form of interaction in which two (or more) agents, with conflicting interests and a desire to cooperate, try to reach a mutually acceptable agreement. Negotiation between two agents can in many ways be modeled as a game, and game theory is useful to analyze the behavior of negotiating agents. Negotiation is, however, also different from many board games such as chess and reversi. One of the most important differences is that negotiation as we will study it in this practical assignment never is a zero-sum game. That is, a typical negotiation does not have a winner who takes all and a loser who gets nothing. In order to start a negotiation, it is only reasonable for both parties to believe that there is a *win-win* situation where both agents can gain by obtaining a deal through negotiation. Another difference is that the domain of negotiation (what the negotiation is about) may be quite different from one negotiation to the other.

To clarify, we add some remarks about the *process* of negotiation. Typically, negotiation is viewed as a process divided into several phases. Initially, in a *prenegotiation phase* the negotiation domain and the issue structure related to the domain of negotiation are fixed. Negotiation may be about many things, ranging from quite personal issues such as deciding on a holiday destination to strictly business deals such as trading orange juice in international trade. In this assignment, the rantal house domain has been selected and the structure of this domain is provided to you. In other words, the prenegotiation phase has been completed and you cannot redefine the domain anymore. There are two tasks that are usually considered part of the prenegotiation phase that you do need to consider in this assignment. The first task involves creating a so-called *preference profile* that captures your own preferences with regards to the rental house domain. The result will be a formal preference function that maps each possible *outcome* of a negotiation to a *utility number* in the range of 0 to 1. The second task involves thinking about a *strategy* used to perform the negotiation itself. But the most important part of this assignment concerns the *negotiation phase* itself, i.e. the exchange of offers between you (or your software agent) and an opponent. Figure 1 provides some initial guidelines based on human experience with negotiation that may help you during your own negotiations and in building your own negotiating software agent.

For this assignment, you will be building your own negotiating agent for a multi-party negotiation. There can more than one opponent in the negotiation. This will be a *team effort* with each team consisting of 2 students. As a team you will design and implement your negotiating agent in Java to do negotiations for you.

Some techniques relevant for building and designing negotiating agents can be found among others in chapters 16 and 17 in [12]. The assignment will also require you to go beyond the course material in the book. Additional information is provided to you in the form of several papers [1, 8, 11, 5, 13]. There is a lot of other literature available about negotiation that may help you finish this assignment successfully. As

Orient yourself towards a win-win approach.

Plan and have a concrete strategy.

Know your reservation value, i.e. determine which bids you will never accept.

Create options for mutual gain.

Take the preferences of your opponent into account.

Generate a variety of possibilities before deciding what to do.

Pay a lot of attention to the flow of negotiation.

Figure 1: Negotiation guidelines

for almost any subject, you can find more information about negotiation strategies online. We recommend to search for strategies used in the ANAC competition [2, 3, 4, 6, 7, 9, 10, 14].

2 Assignment Description

2.1 Objectives

- To learn to design a negotiating agent for a realistic domain with both discrete and integer issues, including among others a negotiation strategy.
- To learn techniques for implementing (adversarial) search and design heuristics while taking into account time constraints.
- To actively interact with other students and participate in student groups by discussing and coordinating the design and construction of a negotiating agent.

2.2 Deliverables

A negotiating agent programmed in Java for a multi-party negotiation using the negotiation environment provided to you. Remember to submit all the relevant source files.

Please make sure all your files are in the directory structure as shown below. Assuming that the group has number three, the directory structure would look like:

```
negotiator/
    group3/
        Group3_Agent.class
        Group3_Agent.java
        SomeHelperClass.class
        SomeHelperClass.java
```

2.3 Requirements

- The agent should implement a negotiation strategy to generate offers, and an acceptance strategy to decide whether to accept an offer or not.
- The agent must aim at the best negotiation outcome possible (i.e. the highest score for itself given the agent's preferences, while taking into account that it may need to concede to its opponent).

- The agent must take the utility of the opponents into account. Of course, initially you only have information about your own preferences. You will have to *learn* the preferences of the opponents during the negotiation process. The default way is by constructing an *opponent model* to have an idea of the utility of your proposed bids for the opponent. We already provided an opponent model, which you are asked to improve.
- The agent meets reasonable time and memory constraints. Any agent 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 agent is able to work on at least all given scenarios. You may assume that the domains solely contain *discrete* or *integer* issues. You may also find references in the negotiation environment to *reservation values*.
- The agent implementation should be tested to ensure that it works on multiple systems, and requires nothing other than the files contained in your group's package. Integrating your agent into the negotiation environment should not be more difficult than adding your package (e.g. the folder "negotiation/group*n*/") in the right folder (the same directory in which the negotiation simulator `negosimulator.jar` can be found).

As a final remark, we want to make clear that it is very important to test your agent in order to improve the negotiation strength of your agent. A good outcome is determined by other criteria, identified by efficiency of the outcome. To test your agent in this regard, it will in particular be useful to test your agent on various negotiation templates. You can create these templates both by modeling real-life examples of negotiation or create them randomly.

3 Organization

You should e-mail the instructor to submit deliverables to the assignment or ask questions.

Important Dates:

1. Deadline D1 September 1, 2015: Deadline for registering your team (team size: 2) and submit the preference profiles. We will assign team numbers shortly after that.
2. Deadline D2 September 29: Submit a functional agent that beats the random agent.
3. Deadline D3 October 24: Submit the improved agent.
4. Deadline D4 November 3: Submit the report and the final agent.

Please submit your report in PDF format and the package for your negotiating agent on moodle. Use the naming conventions described in Sec. 2.2, 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. If you have problem with your agents, please contact us in advance.

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 the design of your negotiating agent; Quality and completeness of answers and explanations provided to the questions posed in the assignment; Explanatory comments in source code, quality of documentation,

- *Performance:* Agents will be ranked according to negotiating strength that is measured in a tournament (see also the next section below),
- *Originality:* Any original features of the agent or solutions to questions posed 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 agents or other students will not be evaluated and will be heavily penalized.**

4.1 Competition

Agents of teams will be ranked according to negotiation strength. The ranking will be decided by playing a tournament in which every agent plays negotiation sessions against a set of agents – including the agents programmed by your peers – on a set of domains. Therefore, your agent should be generic enough to play on any domain.

Agents may be disqualified if they violate the spirit of fair play. In particular, the following behaviors are strictly prohibited: designing an agent in such a way that it benefits some specific other agent, starting new Threads, or hacking the API in any way.

The components that are graded and their relative weights are described in Table 1 below.

Grading Method	**Weight**
D1 (Registration of groups + Submission of preference profiles)	10%
D2 (Genius installed + Simple agent that wins against random agent + Explanation of results)	(10% + 10% + 5%)
D3 (Improved agent)	TBD
D4 (Tournament + Report with design of agent and viva)	(30% + 35%) - (D3 weightage)

Table 1: Grading criteria and weight

For the interested teams, it maybe possible to extend your agent to participate in the ANAC 2017 competition depending on the new problem statement.

5 Acknowledgements

Preparation of this assignment involved the help of many people, including R. Aydogan, T. Baarslag, D. Festen, C. Rozemuller, L. van der Spaa, B. Grundeken, M. Hendrikx, K. Hindriks, C. Jonker, W. Pasman, D. Tykhonov, and W. Visser. We would like to thank them for providing us with an earlier version of the document which we revised for our purposes.

References

[1] Tim Baarslag, Koen Hindriks, Mark Hendrikx, Alex Dirkzwager, and Catholijn 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 = http://mmi.tudelft.nl/sites/default/files/boa.pdf.

[2] Tim Baarslag, Koen Hindriks, and Catholijn Jonker. A tit for tat negotiation strategy for real-time bilateral negotiations. 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 229–233. Springer Berlin Heidelberg, 2013.

[3] 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.

[4] 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.

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

[6] Radmila Fishel, Maya Bercovitch, and Yaakov(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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

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

[14] 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.