Practical Assignment for Multiagent Systems: Designing a Negotiation Agent

February 16, 2024

Abstract

This practical assignment is about designing a negotiation agent in NEGMAS. For questions regarding this practical assignment, consult the FAQ information and/or visit the instruction hours.

1 Introduction

Imagine you have just graduated and plan to throw a nice party for the occasion. The problem is that you do not want to organize everything yourself. Fortunately, a friend of yours has also graduated and wants to throw a party as well. You decide to organize the party together; however, you have different preferences about essential aspects of the party, such as the choice of food, drinks, music, etc. You will now have to make joint decisions about what the party will look like; that is, you will have to *negotiate*. Being a computer science student you use your skills to make a negotiating agent so the party is as close to *your* preferences as possible.

Figure 1: Organizing a party involves choosing music, food, drinks, catering, etc.

This assignment is about *negotiation*: 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 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 almost 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 possible 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. 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 party domain serves as an example and the structure of this domain is provided to you. The party domain is about co-organizing a party and negotiating what you want to spend the available money on. This is set; you cannot redefine the domain anymore. Moreover, we created a so-called *preference profile* that captures the agent's own preferences with regards to the party domain. The result is a formal preference function that maps each possible *outcome* of a negotiation (e.g. a party with rock music, catering, and handmade cocktails) to a *utility value* in the range of 0 to 1. This is set in advance as well. However, there is one task that you do need to focus on in this assignment. This involves thinking about a *strategy* used to perform the negotiation itself. In negotiation you need at least two strategies: an offering strategy (what to bid when), and an acceptance strategy (when to accept or reject offers, and when (if ever) to stop negotiating - walk away). 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 the opponent.

Figure 2 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.

Think of a good starting bid.
Plan and have a concrete strategy.
Know your reservation value, i.e. determine which bids you will never accept.
Take the preferences of your opponent into account.

Figure 2: General negotiation guidelines

This assignment concerns designing and building your own negotiating software agent. This will be a *team effort*: as a team you will design and implement your negotiating agent in Python 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 [1], chapter 2.3 in [2], [3], and [4]. You might also want to go beyond the standard course material. Additional information is provided to you in the form of several papers [5, 6, 7, 8, 9]. 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 [10, 11, 12, 13, 14, 15, 16, 17].

The remainder of this document is organized as follows. Section 2 provides some background to the assignment. In Section 3, you are introduced to the platform and associated tutorials, and receive further details on the assignment.. In Section 4 the objectives, deliverables, requirements and submission details are described. Section 4.4 describes some organizational details and important dates, including deadlines. Finally, Section 5 documents the evaluation criteria and grading for this assignment.

2 Background

Before we present the main tasks and questions you need to complete, some additional background is provided that may help you complete this assignment successfully. After that we present some easy steps to help make yourself familiar with the negotiation environment [18]. Part of that requires you to start

and use the negotiation environment and read the user guide as well as this assignment provided to you carefully. Note that the negotiation environment is a scientific software tool which is being improved constantly. Please report any major bugs found so that we can resolve them in a next version.

A negotiation in this assignment is also called a *negotiation session*. In a session two agents negotiate with each other to settle a conflict and negotiate a deal. Each agent has a private **reservation value**, which is the value that the agents gets if the negotiation fails, which remains constant throughout the course of a session. Each negotiation session is limited by a fixed amount of time. At the end of a session, a score is determined for both agents based on the utility of the deal for each agent if there is a deal, otherwise the score equals the *reservation value*. In a sense, a negotiation does not yield a winner since each agent will obtain a score based on the outcome and its own utility function. A failed negotiation, in the sense that no deal is reached, thus is a missed opportunity for both agents. Your agent will also negotiate in a tournament with all other agents where the scores of each session are recorded and averaged to obtain an overall score (see Section 5). A ranking will be compiled using these overall scores.

Each negotiator will have a specific set of *preferences* regarding the possible outcomes. Such preferences can be specified by an ordering over the set of possible outcomes: an outcome ω is said to be weakly preferred over an outcome ω' by an agent if $\omega \succeq \omega'$, or strictly preferred if $\omega \succ \omega'$. Under mild assumptions [4], preferences can also be expressed in a more complete, numerical way by a so-called *utility function* (cf. [1]). In our case, utility functions assign quantitative values to bids in the negotiation space. In this assignment, you may assume that all utility functions are additive, i.e. they will always be linear in the number of issues. For example, if there are four issues to negotiate about, the utility function can be computed by a *weighted sum* of the values associated with each of these issues. So, let $bid = \langle i_1, i_2, i_3, i_4 \rangle$ be a particular bid. Then the utility $u(bid) = u(i_1, i_2, i_3, i_4)$ (given weights w_1, w_2, w_3, w_4) can be calculated by:

$$u(i_1, i_2, i_3, i_4) = w_1 \cdot u(i_1) + w_2 \cdot u(i_2) + w_3 \cdot u(i_3) + w_4 \cdot u(i_4).$$

Key to negotiation is the fact that you have incomplete information about your opponent. You may assume that there is a conflict of interests, but at the start you do not know on which issues agents agree and on which they disagree. For example, in a negotiation about buying a laptop, the buyer may prefer to have a middle-sized screen but the seller may prefer to sell laptops with small screens because they have more of those in stock. They could, however, find themselves aligned on what brand of laptop that they want to buy/sell. In this assignment, the negotiation task is made a bit easier, since **the utility function of the opponent is given**. However, *the reservation value is still unknown*, which leaves a challenge for the agent.

Ideally, an agreed solution has certain properties. One of these properties is that the outcome should be Pareto optimal. An outcome is Pareto optimal when there does not exist a deal where one agent can do better and the other can do no worse (i.e. they both score higher or equal, and therefore would prefer this new deal over the old one). Another property is related to the Nash solution concept. A Nash solution is an outcome that satisfies certain bargaining axioms and may be viewed as a "fair outcome" which is reasonable to accept for both parties. An outcome is said to be a Nash solution whenever the product of the utility (in the range $[0, 1]$) of the outcome for the first agent and that for the second agent is maximal (cf. [9]; note how this is different from the concept of a Nash equilibrium).

The notion of a fair outcome is important in negotiation because it provides a reference point for what your opponent might be willing to accept. Typically, a negotiation is started in the first place because reaching an agreement is preferable for both. But in order to get an agreement, you will need to get your opponent to agree. In a negotiation between self-interested agents, however, each agent is determined to get the best possible outcome for itself.

The Nash solution concept excludes certain outcomes as being unfair. It is, for example, very unlikely that your opponent will accept an offer that is most favorable to you and leaves your opponent with empty hands. Therefore, an outcome is usually the result of a number of concessions that both parties make. The speed of making concessions, or the concession rate, is the derivative of the (size of the) concession steps taken during a negotiation. Concessions can be made in various ways; an agent that makes concessions in very small steps initially uses a so-called Boulware strategy. A strategy that makes faster concessions initially is called a Conceder. Other strategies are conceivable and may be necessary given the deadlines (cf. [19]).

The outcome space of a negotiation, i.e. all possible bids, can be plotted on a graph which indicates the utility of both agents on the x and y axes respectively. An example is provided in Figure 3. This graph is useful for understanding and analyzing bilateral negotiations. This will become clear while you work on this assignment.

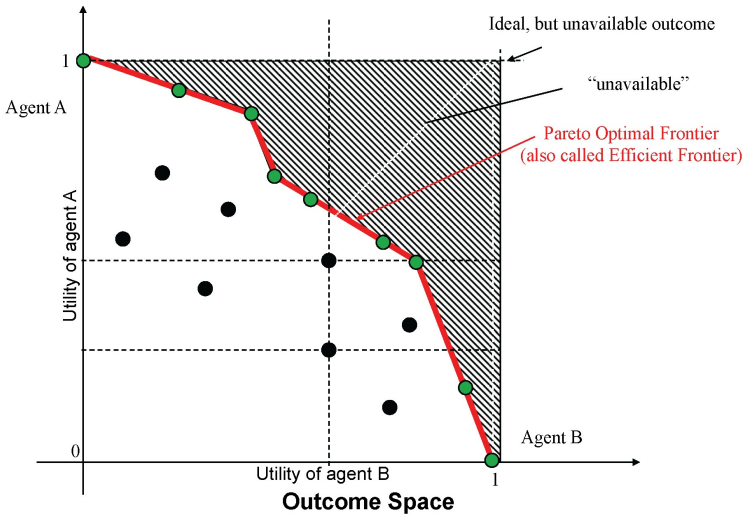

Figure 3: Example of an outcome space for two agents A and B with the Pareto Frontier labeled in red

Besides utility, often in negotiations time plays a crucial role. Negotiations can have deadlines, e.g., if the date of the party has already been fixed, then an agreement should be made in advance of that day. In this assignment there is a deadline for reaching an agreement: both parties have exactly the same deadline for achieving a deal and both parties "know" this. If they fail to reach an agreement each participant gets the reservation value.

3 Assignment

The assignment consists of first familiarizing yourself with the negotiation environment and then designing and implementing a negotiating software agent. You do not have to discuss the questions in the tutorials of 3.1 in your report. This assignment is based on the Automated Negotiations League (ANL) of the Automated Negotiating Agents Competition (ANAC) 2024. The same tutorials and documentation as used for the competition are available for you, as well as extensive documentation of the platform NEGMAS.

3.1 Familiarizing Yourself With the Negotiation Environment

i. Go to the ANL website to find instructions to install the package ANL (and the corresponding platform NEGMAS). You can for example use PyCharm to do so.

ii. Follow the tutorials on the ANL website. If you wish, you may download the code that corresponds with the tutorials here or via this link `http://tinyurl.com/ANAC2024docs`.

   (a) Running a negotiation.

   (b) Develop a negotiator.

      * The part "A more meaningful negotiator" is a recommended advanced section that can be used for inspiration and further insight.

iii. Prepare yourself for the final assignment by downloading the code template (which is called the `ANLSkeleton`) here or via this link `http://tinyurl.com/ANAC2024docs`. This will form

the basis for your own agent. Further help with coding can be found in the ANL documentation, and more detailed in the NEGMAS documentation.

3.2 Designing and Implementing a Negotiating Agent

The remainder of this assignment concerns the design and implementation of a negotiation agent. The challenge of this year is:

> Design a negotiation agent for bilateral negotiation that has access to its own utility and its opponents utility, but not its opponent's reservation value.

This negotiating agent will be tested on a large set of scenarios. Code to test your agent on examples of these scenarios is included in the provided template.

If enjoy thinking about efficient negotiating strategies, please refer to the last part of this assignment (Sec. 6) for information on how to do a MSc. thesis about negotiation agents.

1. PREPARATION: ANALYZE NEGOTIATION DOMAIN

   (a) Consider the following holiday negotiation between two agents *A* and *B*:

Issue:	Possible Values:
Location	Antalya, Barcelona, Milan
Duration	1 week, 2 weeks
Hotel Quality	Hostel, 3 star hotel, 5 star hotel

Agent A's preferences:

Weights: W_{location}= 0.5 W_{duration}= 0.2 $W_{\text{hotel-quality}}$=0.3
Evaluation values: 4, 10, 2 (for Antalya, Barcelona and Milan respectively)
Evaluation values: 3, 10 (for 1 week and 2 weeks respectively)
Evaluation values: 10, 2, 3 (for hostel, 3 star hotel and 5 star hotel respectively)

Agent B's preferences:

Weights: W_{location}= 0.5 W_{duration}= 0.4 $W_{\text{hotel-quality}}$=0.1
Evaluation values: 3, 2, 10 (for Antalya, Barcelona and Milan respectively)
Evaluation values: 4, 10 (for 1 week and 2 weeks respectively)
Evaluation values: 3, 3, 10 (for hostel, 3 star hotel and 5 star hotel respectively)

Table 1: Description of the Holiday negotiation scenario

   Use the provided code in NEGMAS to define the scenario and to compute and draw the outcome space (without running the negotiation) and indicate the Pareto efficient frontier in a graph. Hint: Use the tutorial "Running a negotiation" or look at the NEGMAS documentation.

   (b) Run the provided code for assignment B to load **the party domain.** Add a reservation value of 0.5 to the utility function of agent A and 0.2 to the utility function of agent B.[1]. Let agent A be played by the `Awesome Negotiator` agent imported from the template, and let agent B be played by a `Boulware` agent, and run several negotiations on the party domain. Again, use NEGMAS to draw the Pareto frontier in a graph (including indication of `reserved` values!) of one of your runs, and visualize the bidding curves of both agents. How are the bidding curves influenced by the reservation value?

[1] Note, add the reservation value in the code after normalizing the utility function. In NEGMAS, reservation values are called *reserved values*

2. Design a negotiation strategy

In this section you have to design and implement your negotiation agent using the BOA framework. A main advantage of this framework is that you can benefit from using a large set of existing components and mix and match them [20]. Before doing so, it is recommended to read about existing negotiation strategies (cf. [21, 11, 12, 13, 14, 15, 16, 17]).

Note that this section adheres to the good practice of designing an agent properly before implementation; it is allowed however, and even stimulated, to later amend your design when you have new ideas or when additional evidence turns up (e.g. during the evaluation in Section 5), as long as these changes in design are documented and substantiated in your report.

(a) A PEAS description [1] reports on an agent in terms of performance, environment, actuators and sensors:

- *Performance measure*: how do you measure the performance of your agent?
- *Environment*: to specify the task environment, it suffices to provide a detailed account of this negotiation assignment setup. What setting your agent will face (i.e. what is there)?
- *Actuators*: what actions are available to your agent (i.e. what can be done)?
- *Sensors*: the 'sensor' of a virtual agent is the API through which it can sense input about its environment. What input is available to your agent (i.e. what can be observed), and what is not?

We provide a PEAS description of the agent for this assignment in the following table, with the performance measure to be filled in:

Performance	***To be filled in.***
Environment	- The opponent - Domain - Protocol - Preferences
Actuators	- Accept - Make bid - End negotiation
Sensors	Through the API, get: - The current opponent move - Bid history - Current time - Own utility

Provide the performance measure for the agent that you need to design and implement in this assignment. Keep in mind that the ultimate performance of the agent is measured in utility, but there may be other performance measures defined by your group that you would like the agent to employ. That is, it is important to think carefully about how *you* will measure the success of your agent in order to achieve the ultimate goal of the agent: to outperform your peers in a tournament. What you would like your agent to aspire to (i.e. what should be achieved)?

(b) For implementing the agent we use the BOA agent framework as introduced in [5] and described next in Exercise 3. Provide a high-level description of your envisioned design. Also discuss each BOA component for your agent separately: The acceptance strategy, the bidding strategy and the reservation value model. Discuss which parts will build on existing techniques and which parts will be new. Argue why you choose the existing techniques and how your ideas add on to it, and think of how you can test them.

3. Implement a negotiation strategy

(a) Implement the **acceptance strategy** Describe which factors the agent takes into account for making this decision. Does your agent take time and/or opponent's actions into account when deciding to accept an offer? In case your agent also uses other considerations to determine whether it will accept particular bids, also explain these considerations.

Useful literature: [22],[23], [24].

(b) Implement the **bidding strategy** Decide on the way your agent will make its first offer and counter offers. Explain why you have chosen this to be the agent's opening bid and how your agent computes a counter offer. List all considerations that your agent takes into account explicitly and explain why you have chosen to base the agent's decision on these considerations.

Does your agent ever propose offers that *increase* the utility for itself relative to its previous bid? If it does, explain when and why it will decide to do this. If not, demonstrate that the utility associated with bids in any bid sequence your agent will ever propose in a negotiation monotonically decreases. If it does, explain when and why it will decide to do this.

Useful literature: [25], [26], [27].

(c) Implement the opponent model for the **reservation value.** For this assignment, you can access the utility function of the opponent. Only the reservation value is unknown. The opponent's behavior could give some hints about it's private value. Explain how your agent tries to learn the reservation value of the opponent.

Useful literature: [28], [29], [30], [31]

4. QUANTIFY THE PERFORMANCE OF YOUR AGENT

(a) **Basic test on Party domain**
Have your agent negotiate against itself on the party domain, and against two of the default ANL competitor agents provided in NEGMAS[2]. Run several negotiation sessions with different preference profiles. Report on the outcomes reached. Try to explain why the outcome is as it is. Are any of the outcomes a Nash solution? Is it possible to reach an efficient outcome, i.e. an outcome that lies on the Pareto Frontier?

(b) **Test on other domains**
One important question remains. How generic is your agent? That is, does it work as well on the party domain as on other domains? To verify this, have your agent negotiate against itself, and against at least two agents provided to you using the standard ANL tournament settings (or a variation of the settings)[3]. Report on the outcomes reached. Try to explain why the outcome is as it is. Is any of the outcomes a Nash solution? How often can your agent reach an efficient outcome, i.e. an outcome that lies on the Pareto Frontier?

(c) **Test your opponent model**
Test your agent in a small tournament and compare your agent with your opponent model for the reservation value, and without. Does your model lead to a significantly higher utility?

Useful literature: [32], [33].

3.3 Concluding: Future Perspectives

5. 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?

Useful literature: [34], [35]

[2] RVFitter, NashSeeker, MiCRO, Boulware, Conceder, Linear

[3] You may use the template for this, and edit it by adding the specific opponents you want to add. You may tweak other parameters as well, such as number of outcomes and the type of generated scenarios (arbitrary_pie_scenarios, monotonic_pie_scenarios, zerosum_pie_scenearios).

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 in different types of settings. A warning is in place: do not assume that your goal is to outperform a 'stupid' agent'. Try to beat as many agents as you can and older versions of your own agent. A good outcome is determined by other criteria, identified by efficiency of the outcome (i.e. is it close or not to the Pareto Frontier). 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.

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.

If your agent performs well enough, you could even participate in the yearly competition for automated negotiating agents, called ANAC[4] [10, 36, 37, 38, 39, 40]. 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.

4 Detailed Assignment Deliverables

The assignment must be completed in teams of 5 students. In the following paragraphs the objectives, deliverables, requirements, and the detailed assignment description are documented.

4.1 Objectives

- To learn and apply techniques from multi-agent negotiation, preference modeling and utility theory, communication protocols and coordination in multi-agent systems, group decision-making, opponent modeling, decision-making under uncertainty.
- To learn to design and analyze a negotiating agent for a realistic domain with discrete 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.

4.2 Requirements

Negotiation agent

You design a negotiating agent programmed in Python using the negotiation environment provided to you.

Agent 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).
- Your opponent model has to *learn* the reservation value of the opponent during the negotiation process.

[4] For more information on the automated negotiation competition see: https://web.tuat.ac.jp/~katfuji/ANAC2024

- 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 party works on all domains with linear additive profiles with *discrete* issues. You may also find references in the negotiation environment to *integer* issues, *real* issues, and *discount factors*, but you can ignore these, as they do not play play a role in this assignment.

Code requirements:

- Your submission should consist of a package containing your source agent code, that is based on the template agent provided to you.
- Rename "myagent" folder and "myagent" file to "group" + *n*, e.g "group3". Rename the class of your agent in the code to "Group" + *n* (note the capital!).
- If you import files in your agents code, makes sure you turn them into packages, by adding empty `__init__.py` files to every folder you want to import from. For more info, see the FAQ.
- 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 "group*n*/") in the right folder.
- The source code should contain explanatory comments that will allow third parties 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 agent does not print anything to the screen when handed in.

Please make sure all your files are in the directory structure as explained above. Assuming that you are group number 3, a valid directory structure is (based on the template):

```
group3/
    helpers/
        _init_.py
        runner.py
        somehelperclass.py
        somehelperclass.py
    report/
        Group3_final_report.pdf
    _init_.py
    group3.py
    README.md
```

This whole directory structure should be send in a **ZIP** file.

Report

You hand in reports documenting and explaining the solution. See Table 2 for an overview of which questions to cover in each report.

Submission requirements:

- Include your group number in every report.

- The final report need not be lengthy (5 pages may be enough, 9 pages maximum, not counting references), but should include an explanation and motivation of *all* of the choices made in the design of negotiating agent. The use of figures that support your story is encouraged. References do not count toward the page limit.
- 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 Python functions used by your agent should be explained in the report itself.
- An advised (but not obligatory) structure of the report is:
  - (Include your group number on top of the report);
  - An introduction;
  - A high level overall description of the agent;
  - A description per component (bidding, acceptance, opponent model)
    - A high-level description of the agent and its structure, including the main python functions (mention these explicitly!) used in the negotiating agent that have been implemented in the source code. This includes detail of any changes that were made to the original agent design and a motivation for these;
    - An explanation of the negotiation strategy, decision function for accepting offers, any important preparatory steps, learning techniques, and heuristics that the agent uses to decide what to do next;
    - Include the answers to question 3.
  - Quantifying the agent's performance
    - A section documenting the strong and weak points of your agent, the tests you performed to analyze (and improve) the negotiation strength of your agent. 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 agent, and motivate why you choose these testing methods/metrics/graphs;
    - Include the answers to question 4.
  - Future perspectives
    - Include the answers to question 5.
  - Conclusion
    - Include a final reflection written by the group coordinator, in which the coordinator discusses a summary of the experience of the team with regards to writing the report and building the negotiating agent, and a summary of who performed which tasks.

4.3 Rules

Fair Play

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.

Plagiarism

Both the agent source code and the report need to be your own work. For the agent, you can use the code from example agents provided, but anything else needs to be clearly acknowledged in the report and when submitting the agent. Note that you are not allowed to directly use code of agents programmed by others or external tools. 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.

Deadline	Topic	Deliverables
15 Feb 2024, 23:59 CEST	Group registration	Group members and group coordinator
1 Mar 2024, 23:59 CEST	Question 1-2	First report (3 pages max) Note, question 2b is the most important.
28 Mar 2024, 23:59 CEST	Question 3-5 + negotiation agent	Final report: adhering to the structure (5 pages may be enough, 9 pages maximum, not counting references) and expected content mentioned in the requirements section. Source code with correct packaging and structure.

Table 2:

4.4 Submission

Only group coordinators may use the submission email **tamara.florijn@cwi.nl** to submit deliverables to the assignment. Make sure to read the requirements section thoroughly before working on your deliverables.

Important Dates:

1. **Group deadline**: Deadline for registering your group by the group coordinator is **15 Feb 2024, 23:59 CEST**. Please register as a group through e-mail before the deadline. Make sure your group consists of individuals with a heterogeneous background, with a range of expertise that covers all aspects of the assignment.

   Once the deadline has passed, it is not possible to change groups. All subsequent assignment submissions by the group coordinator are considered as submissions from the group. Should any problems arise within the group, the group coordinator should get in contact with the course managers well before the submission deadline.

2. **Submission deadlines**: There are 2 intermediate submissions and one final submission for this assignment. The deadlines and expected deliverables for all the submissions are given below. Please note that compilation failures and execution errors will be perceived as an incomplete submission.

The group coordinator submits the reports in PDF format and the package for your negotiating agent through e-mail. 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. If you have problems with your assignment, please contact us well in advance. Note that the last opportunity for this is the Question & Answer session on **28 Mar 2024 (15:15-17:00)**.

5 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 final grade for the tutorial will be determined as a weighted average of several components, based on your team solution for your reports and the quality and performance of your negotiating agent. The components that are graded and their relative weights are described in Table 3 below.

Assignment	**Weight**
First report	20%
Final report and Negotiation agent	80%

Table 3: Grading weights for each deliverable

Broadly, the assignments 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 and quality of documentation.
- *Performance:* Agents of teams will be ranked according to negotiation strength (i.e. average selfish utility obtained in all settings). 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 similar to the ones you have encountered in this assignment. Therefore, your agent should be generic enough to play on any domain.
- *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 be judged as insufficient. Detection of fraud will be reported to the administration.

The final report will be graded according to the set of measures outlined in Table 4.

Criterion	**Measure**	**Weight**
Description	Completeness, correctness and clarity of the agent description	10%
Strategy	Motivation and understanding about the strategic aspects of the negotiation game	15%
Creativity	Sophistication and originality of the agent design through novel ideas and adaptations	15%
Literature	Motivation by, support from, and improvement over existing literature	10%
Analysis	Discussion of the performance of the agent, and ways to overcome the deficiencies	20%
Agent performance	Performance in a tournament with other agents, including baseline agents and agents implemented by other groups	20%
Code quality	Code correctness, readability, and class composition	10%

Table 4: Grading criteria, measures, and weights.

6 Master Thesis about negotiation

Did you like thinking about efficient negotiating strategies, or implementing a successful negotiating agent? Automated negotiation provides a lot of subjects to do your Master Thesis on! You can always ask the course assistants, have a look at the last page of this assignment, or contact us for more information: **T.Baarslag@uu.nl**.

Acknowledgements

This assignment has been written with the help of many people, including R. Aydoğan, T. Baarslag, C.P. Florijn, H. Griffioen, C.M. Jonker, W. Pasman, P. Prajod, Y. Mohammad.

Interested in the possibility of doing a Master Thesis about negotiation?

Negotiation is a major research area of AI and one of the main tools an autonomous agents to agree about their course of action. As such, you might be interested in doing a Master Thesis about negotiation. Contact **T.Baarslag@uu.nl** for more information or see https://uu.konjoin.nl/project/1183-negotiation-ai for possible projecs.

Negotiation has become a major and interesting research area within Artificial Intelligence. Computers that are able to negotiate are already common practice in areas such as high frequency trading, and are now finding applications in domains closer to home, such as the smart grid, market places, real estate, the Internet of Things, and autonomous vehicles. These negotiations involve not just financial optimizations but balanced trade-offs between multiple topics, such as cost and convenience. Such agents can autonomously negotiate and coordinate with others in our stead, to reach outcomes and agreements in our interest.
There are many angles that you can take, e.g.:

- Design of negotiation *strategies* and better *opponent models* (e.g. using machine learning techniques);
- Design methods to negotiate with *multiple opponents* at once, in settings such as procurement and trading. Here, we need to search for smart *combinations* of multiple deals.
- Design of a *negotiation support system*, either advising humans in a particular domain, or even taking over negotiation all together;
- Design of a negotiation decision model that can capture and reason about *uncertain preferences* about the user and can query the user for more preference information? In such cases, the agent needs to strike a balance between increasing the user model accuracy and the inconvenience caused by interacting with the user.
- Design of a *negotiation protocol* that can generate better agreements (e.g., multi-agent settings, auctions, use a new form of communication between the agents);
- Design of a *testbed* for comparing negotiating agents;
- Extend the current negotiation setting to more expressive *preferences*, for instance non-linear utility functions with dependencies between different issues;
- Design agents in a *multi-agent many-to-many setting*, for example an auction;
- Research related to *choosing* your negotiation partner, for instance trust and the reputation of agents;
- etc.

This can all be combined with a Literature Survey, in order to kick-start your thesis. An eligible candidate has good mathematical skills, and a passion for iterative research and design: i.e. come up with of a solution, try it out theoretically or in simulation, and re-assess in an iterative fashion. As a MSc student, you have the chance to have an office and desk both in Utrecht and the CWI research institute in Amsterdam, with the opportunity for a small monthly stipend for excellent students (average grade > 8) who like to stand out during their thesis project.

If you are interested, don't hesitate to ask us: **T.Baarslag@uu.nl**

References

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2010.

[2] Y. Shoham and K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009. [Online]. Available: http://www.masfoundations.org/mas.pdf

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

[4] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives*. Cambridge University Press, 1976.

[5] T. Baarslag, K. V. Hindriks, M. J. Hendrikx, A. S. Dirkzwager, and C. 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. [Online]. Available: https://repository.tudelft.nl/islandora/object/uuid:2d8f71a7-9311-43d9-8ba0-b418a97aab7f/datastream/OBJ/download

[6] C. M. Jonker and J. Treur, "An agent architecture for multi-attribute negotiation," in *Proceedings of IJCAI'01*, 2001, pp. 1195–1201.

[7] R. Lin, S. Kraus, T. Baarslag, D. Tykhonov, K. V. Hindriks, and C. M. Jonker, "Genius: An integrated environment for supporting the design of generic automated negotiators," *Computational Intelligence*, vol. 30, no. 1, pp. 48–70, 2014. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8640.2012.00463.x

[8] 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. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889098000293

[9] R. Serrano, "Bargaining," 2005. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.113.8981&rep=rep1&type=pdf

[10] T. Baarslag, R. Aydoğan, K. V. Hindriks, K. Fujita, T. Ito, and C. M. Jonker, "The automated negotiating agents competition, 2010-2015," *AI Magazine*, vol. 36, no. 4, pp. 115–118, 12/2015 2015. [Online]. Available: http://www.aaai.org/ojs/index.php/aimagazine/article/view/2609

[11] M. Ben Adar, N. Sofy, and A. Elimelech, "Gahboninho: Strategy for balancing pressure and compromise in automated negotiation," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 205–208. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_13

[12] A. Dirkzwager, M. Hendrikx, and J. Ruiter, "The Negotiator: A dynamic strategy for bilateral negotiations with time-based discounts," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 217–221. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_16

[13] R. Fishel, M. Bercovitch, and Y. Gal, "Bram agent," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 213–216. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_15

[14] A. Frieder and G. Miller, "Value model agent: A novel preference profiler for negotiation with agents," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 199–203. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_12

[15] S. Kawaguchi, K. Fujita, and T. Ito, "AgentK2: Compromising strategy based on estimated maximum utility for automated negotiating agents," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 235–241. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_19

[16] T. van Krimpen, D. Looije, and S. Hajizadeh, "Hardheaded," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 223–227. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_17

[17] C. R. Williams, V. Robu, E. H. Gerding, and N. R. Jennings, "Iamhaggler2011: A gaussian process regression based negotiation agent," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 209–212. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_14

[18] Y. Mohammad, S. Nakadai, and A. Greenwald, "NegMAS: A Platform for Automated Negotiations," in *PRIMA 2020: Principles and Practice of Multi-Agent Systems*, ser. Lecture Notes in Computer Science, T. Uchiya, Q. Bai, and I. Marsá Maestre, Eds. Springer International Publishing, 2021, pp. 343–351.

[19] S. S. Fatima, M. J. Wooldridge, and N. R. Jennings, "Optimal negotiation strategies for agents with incomplete information," in *Revised Papers from the 8th International Workshop on Intelligent Agents VIII*, ser. ATAL '01. London, UK: Springer-Verlag, 2002, pp. 377–392. [Online]. Available: http://dl.acm.org/citation.cfm?id=648208.757345

[20] T. Baarslag, A. S. Dirkzwager, K. V. Hindriks, and C. M. Jonker, "The significance of bidding, accepting and opponent modeling in automated negotiation," in *21st European Conference on Artificial Intelligence*, ser. Frontiers in Artificial Intelligence and Applications, vol. 263. IOS Press, 2014, pp. 27–32. [Online]. Available: http://ebooks.iospress.nl/volumearticle/36911

[21] T. Baarslag, K. V. Hindriks, and C. M. Jonker, "A tit for tat negotiation strategy for real-time bilateral negotiations," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 229–233. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_18

[22] ——, "Effective acceptance conditions in real-time automated negotiation," *Decision Support Systems*, vol. 60, pp. 68–77, Apr 2014. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2013.05.021

[23] T. Baarslag and K. V. Hindriks, "Accepting optimally in automated negotiation with incomplete information," in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, ser. AAMAS '13. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 715–722. [Online]. Available: http://dl.acm.org/citation.cfm?id=2484920.2485033

[24] S. Kawaguchi, K. Fujita, and T. Ito, "Compromising strategy based on estimated maximum utility for automated negotiating agents," in *New Trends in Agent-based Complex Automated Negotiations, Series of Studies in Computational Intelligence*, T. Ito, M. Zhang, V. Robu, S. Fatima, and T. Matsuo, Eds. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 137–144. [Online]. Available: http://link.springer.com/content/pdf/10.1007%2F978-3-642-24696-8_8

[25] R. Ros and C. Sierra, "A negotiation meta strategy combining trade-off and concession moves," *Autonomous Agents and Multi-Agent Systems*, vol. 12, pp. 163–181, 2006. [Online]. Available: http://dx.doi.org/10.1007/s10458-006-5837-z

[26] S. S. Fatima, M. J. Wooldridge, and N. R. Jennings, "Multi-issue negotiation under time constraints," in *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2002, pp. 143–150.

[27] P. Faratin, C. Sierra, and N. R. Jennings, "Using similarity criteria to make issue trade-offs in automated negotiations," *Artificial Intelligence*, vol. 142, no. 2, pp. 205 – 237, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0004370202002904

[28] T. Baarslag, M. J. Hendrikx, K. V. Hindriks, and C. M. Jonker, "Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques," *Autonomous Agents and Multi-Agent Systems*, vol. 30, no. 5, pp. 849–898, 2016. [Online]. Available: http://dx.doi.org/10.1007/s10458-015-9309-1

[29] A. S. Gear, K. Prakash, N. Singh, and P. Paruchuri, "PredictRV: A Prediction Based Strategy for Negotiations with Dynamically Changing Reservation Value," *Group Decision and Negotiation: A Multidisciplinary Perspective*, vol. 388, pp. 135–148, 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7215186/

[30] C. O. B. Yavuz, Č. Süslü, and R. Aydoğan, "Taking Inventory Changes into Account While Negotiating in Supply Chain Management," in *Proceedings of the 12th International Conference on Agents and Artificial Intelligence*, vol. 1. SciTePress - Science and Technology Publications, 2020, pp. 94–103. [Online]. Available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0008976900940103

[31] C. Yu, F. Ren, and M. Zhang, "An adaptive bilateral negotiation model based on bayesian learning," in *Complex Automated Negotiations: Theories, Models, and Software Competitions*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, and T. Matsuo, Eds. Springer Berlin Heidelberg, 2013, vol. 435, pp. 75–93. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30737-9_5

[32] T. Baarslag, M. J. Hendrikx, K. V. Hindriks, and C. M. Jonker, "Measuring the performance of online opponent models in automated bilateral negotiation," in *AI 2012: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, M. Thielscher and D. Zhang, Eds., vol. 7691. Springer Berlin Heidelberg, 2012, pp. 1–14. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-35101-3_1

[33] ——, "Predicting the performance of opponent models in automated negotiation," in *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, ser. WI-IAT '13, vol. 2. Washington, DC, USA: IEEE Computer Society, Nov 2013, pp. 59–66. [Online]. Available: http://dx.doi.org/10.1109/WI-IAT.2013.91

[34] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, M. J. Wooldridge, and C. Sierra, "Automated negotiation: Prospects, methods and challenges," *Group Decision and Negotiation*, vol. 10, no. 2, pp. 199–215, 2001.

[35] T. Baarslag, M. Kaisers, E. H. Gerding, C. M. Jonker, and J. 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*, ser. IJCAI'17, 2017, pp. 4684–4690. [Online]. Available: https://doi.org/10.24963/ijcai.2017/653

[36] T. Baarslag, K. V. Hindriks, C. M. Jonker, S. Kraus, and R. Lin, "The first automated negotiating agents competition (ANAC 2010)," in *New Trends in Agent-based Complex Automated Negotiations*, ser. Studies in Computational Intelligence, T. Ito, M. Zhang, V. Robu, S. Fatima, and T. Matsuo, Eds., vol. 383. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 113–135. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-24696-8_7

[37] C. M. Jonker, R. Aydoğan, T. Baarslag, K. Fujita, T. Ito, and K. 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*, Feb 2017, pp. 5070–5072. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/download/14745/14021

[38] R. Aydoğan, T. Baarslag, K. Fujita, J. Mell, J. Gratch, D. de Jonge, Y. Mohammad, S. Nakadai, S. Morinaga, H. Osawa, C. Aranha, and C. M. Jonker, "Challenges and main results of the automated negotiating agents competition (anac) 2019," in *Multi-Agent Systems and Agreement Technologies*, N. Bassiliades, G. Chalkiadakis, and D. de Jonge, Eds. Cham: Springer International Publishing, 2020, pp. 366–381.

[39] R. Aydoğan, K. Fujita, T. Baarslag, C. M. Jonker, and T. Ito, "ANAC 2018: Repeated multilateral negotiation league," in *Advances in Artificial Intelligence*, Y. Ohsawa, K. Yada, T. Ito, Y. Takama, E. Sato-Shimokawara, A. Abe, J. Mori, and N. Matsumura, Eds. Cham: Springer International Publishing, 2020, pp. 77–89. [Online]. Available: https://www.jstage.jst.go.jp/article/pjsai/JSAI2019/0/JSAI2019_2F1E301/_article/-char/ja

[40] ——, "Anac 2017: Repeated multilateral negotiation league," in *Advances in Automated Negotiations*, T. Ito, M. Zhang, and R. Aydoğan, Eds. Singapore: Springer Singapore, 2021, pp. 101–115.