The Automated Negotiating Agents Competition 2019

Agent Negotiation under Preference Uncertainty

Challenge for ANAC 2019

The challenge for 2019 is to design an agent that can negotiate under preference uncertainty. The idea is that when a negotiating agent represents a user in a negotiation, it cannot know exactly what the user wants due to practical limits on the preference elicitation process.

In previous years of ANAC, and in most literature on automated negotiation, the utility function of the agent is presumed known. Instead, this year, the preferences of the agent will be given in the form of a *ranking* of a limited number of possible agreements.

Website: http://web.tuat.ac.jp/~katfuji/ANAC2019/#genius

Event

The competition takes place during IJCAI 2019, August 10-16 2019, in Macao, China. The prize money for the competition is at least $5000 in total, and several student travel grants will be made available to attend the competition.

Entrants

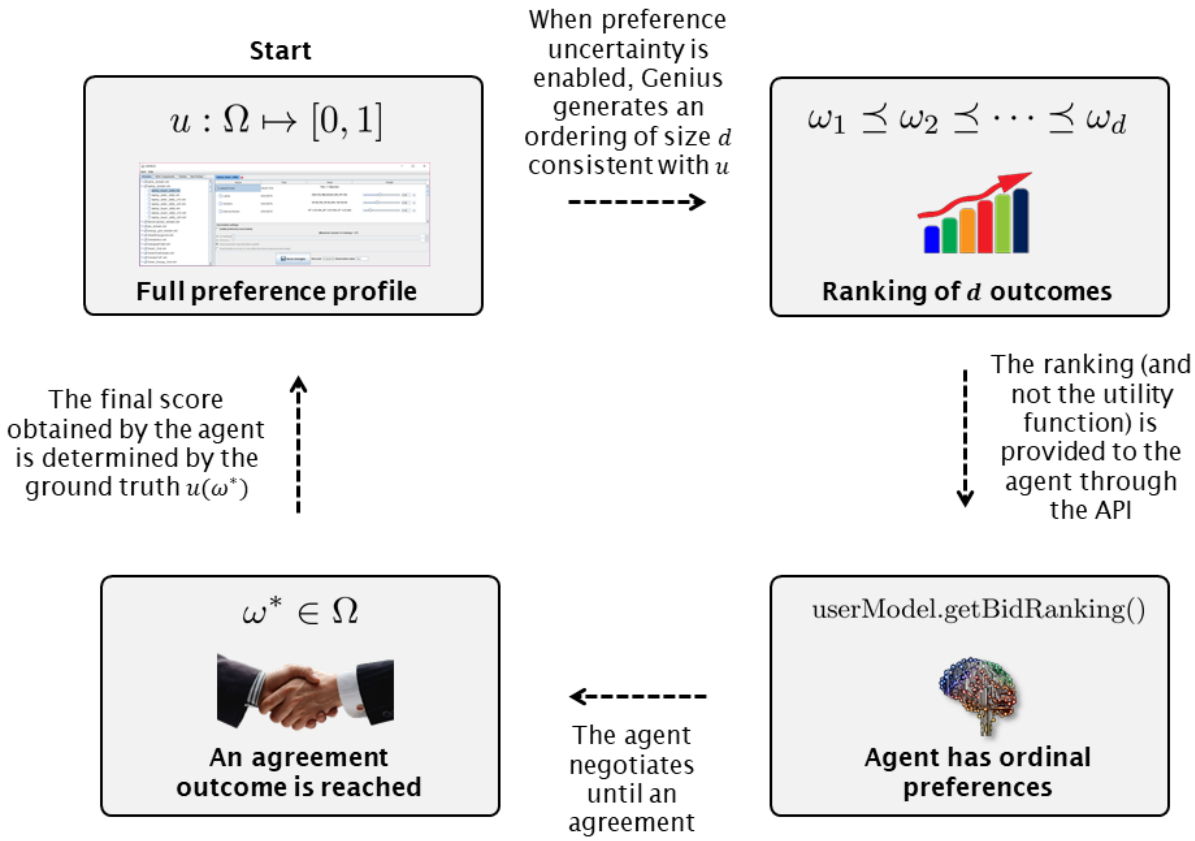
Entrants to the competition have to develop and submit an autonomous negotiating agent that runs on Genius. Genius is a Java-based negotiation platform in which you can develop general negotiating agents as well as create negotiation domains and preference profiles. The platform allows you to simulate negotiation sessions and run tournaments. To aid with development of your agent, a special version of Genius is available containing example agents, domains and scenarios specifically tailored to the setting of ANAC 2019.

Performance of the agents will be evaluated in a tournament between all participants, where each agent is matched with other submitted agents. All submitted agents will negotiate in a number of negotiation scenarios, with varying levels of preference uncertainty.

Negotiating with partial preferences

This year, rather than having access to a utility function, the preferences of the agent will be given in the form of a *ranking* of outcomes (i.e. $\omega_1 \leq \cdots \leq \omega_d$, where each ω_i is a possible agreement).

The rankings are randomly generated from existing negotiation scenarios in which full utility information is available from a standard linearly additive utility function u (see figure below). The number of rankings d that the agent receives represents the amount of preference uncertainty of the agent. The agent negotiates with these ordinal preferences until a certain outcome ω^* is reached. The score the agent will receive for this agreement will be based on the original utility $u(\omega^*)$ that is associated with it. That is, the agent receives ordinal information only, but it will be evaluated based on the underlying cardinal ground-truth.



The challenge for the agents is to estimate the best fitting linear utility function from the ranking, using techniques from e.g. machine learning, regression techniques, trade-offs, and linear programming.

For details, please refer to the Genius manual and the frequently asked questions.

Rules of Encounter

Negotiations are bilateral encounters following the alternating offers protocol. Offers are exchanged using a round-based protocol, with a maximum of 1000 rounds. Agents do not have any prior knowledge about the preferences and strategy of the opponent. Agents are reset after each encounter; that is, agents may negotiate repeatedly on the same domain or with the same opponents, but they cannot learn from their previous negotiations. When no agreement is struck, both agents receive their reservation value, which is privately known utility value between 0 and 1. The reservation value can be different for both players. Note that this means a break-off can be preferable to an agreement for one (or both) of the players. This implies that it is risky to wait until the deadline to reach an agreement. This year features no discount factor.

The alternating offers protocol specifies that both negotiators take turns making offers and is implemented as a special bilateral case of the multilateral Stacked Alternating Offers Protocol. One agent starts the negotiation with an opening bid, after which the other party can take the following actions:

1. Make a counter offer (thus rejecting and overriding the previous offer);
2. Accept the offer;
3. Walk away (i.e. ending the negotiation without any agreement and receiving the reservation value).

This process ends until either an (dis)agreement is reached or 1000 rounds of exchanges have been made. If no agreement has been reached before this time, the negotiation will end in a break-off. In order to make running the tournament feasible, agents are expected to generate their offers in a reasonable amount of time. Agents that take too long will have a time-out imposed on the negotiation after approximately

one minute of real time (the exact moment is not given to avoid strategic considerations around the real-time clock).

Agents will be disqualified for violating the spirit of fair play (e.g. hacking the API, starting threads, attempting to access other party's preference profile). The board of the ANAC 2019 competition will be the judge on these matters (for more information, see http://ii.tudelft.nl/anac/).

Evaluation

The winners will be determined in two separate categories: the average individual utilities gained by each agent, and the average social welfare (i.e., average product of utilities of both agents). The teams of the top performing agents will be notified, and the final results and awards will be announced at IJCAI 2019. It is expected that teams that make it through to the finals will have a representative attending the conference.

Submission (Deadline: 5 June, 2019)

The competition rules allow multiple entries from a single institution, but require each agent to be developed independently. Participants submit their agent source code and class files (in a .zip or .jar package) through the following link: https://tinyurl.com/GENIUSANAC2019

Academic report

Each participant has the option to prepare a 2-4 page report describing the design of heir agent according to academic standards. The best teams that submit a report will be given the opportunity to give a brief presentation describing their agent at IJCAI. Furthermore, proceedings of the competition are planned to be published by Springer in the Studies in Computational Intelligence (SCI) series.

The report will be evaluated by the organizers of the league. For eligibility, the strategy design should provide a contribution to the negotiation community. The report is recommended to address the following aspects:

- **Bidding Strategy:** how the agent generates bid at its each turn;
- **Acceptance Strategy**: how the agent decides to accept or reject a given bid;
- **Opponent Modelling**: how the agent models the opponent (e.g. the opponent's strategy, preferences etc.);
- **Method for dealing with preference uncertainty:** how the agent deals with preference uncertainty, explaining which heuristics and/or machine learning method it employs for this purpose;
- **Evaluation:** an evaluation of the agent (either against itself or in a small-scale tournament setting).

Important Dates

Submission deadline:	June 5, 2019
Notification to finalists:	June 19, 2019 (we intend to inform participants before IJCAI's early registration deadline)
Event:	August 10-16, 2019

Questions and Answers

Website: http://web.tuat.ac.jp/~katfuji/ANAC2019/#genius
Feel free to ask your questions in the FAQ, see the following link: here

Send your questions to:

- T.Baarslag@cwi.nl (main contact)
- reyhan.aydogan@ozyegin.edu.tr
- katfuji@cc.tuat.ac.jp

Quick start guide: how to develop an agent for ANAC 2019

1. Get the latest release of Genius and unzip the contents into a local folder.
2. Import your local folder as a project in your favorite Java IDE. There are tutorial videos available for Eclipse and IntelliJ.
3. Make sure the genius .jar file is added to the classpath of your project and that the example .java files are in the src/ folder. (See the appendix of the user guide for more details.)
4. Run Genius in your IDE (i.e. by running the main **Application.java**). The example agents (i.e. the .java files included in the .zip package) should now compile.
5. You can now import the example agents into Genius through the user interface by choosing to "Add a new party". Point to the .class files (located in the bin/ folder) that have been generated from the example code; i.e.:
    a. **bilateralexamples/RandomBidderExample**
    b. **bilateralexamples/BoaPartyExample**
    c. **bilateralexamples/CustomUtilitySpacePartyExample**
6. Run a negotiation with the **BoaPartyExample** to check if everything is working (e.g. run a negotiation session on the **party_domain** against **BoulwareNegotiationParty**)
7. Make some changes to the code of **BoaPartyExample** (e.g. change its description or the concession parameter *e*).
8. Run a negotiation session with the **BoaPartyExample** again to check if your changes have been reflected.
9. You are now ready to extend the example code further and start building your agent!