Table of Contents

Table of Contents

BRAM Agent: Overview

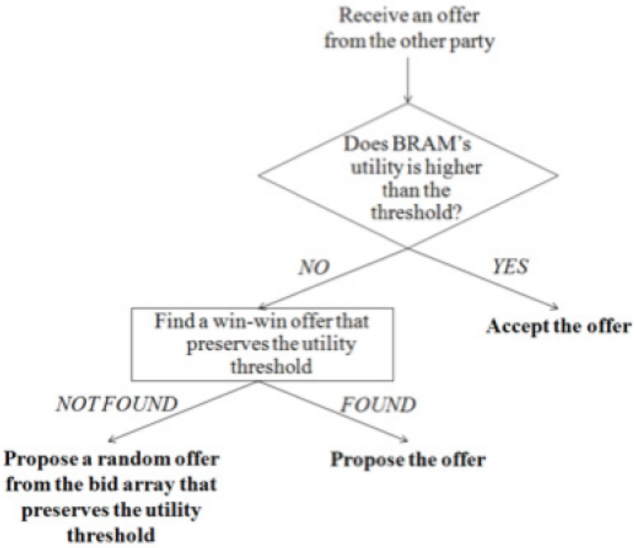
BRAM considers the other agent as a partner in a negotiation process as opposed to an opponent. Its strategy is to reach a solution that satisfies the other agent, while maintaining its own utility threshold. BRAM has four main characteristics:

- Simple and fast
- Compromises as time elapses
- Seeks a win-win agreement
- Interrupts modelling attempts by others (Not mentioned in the paper explicitly)

BRAM Agent: Strategy

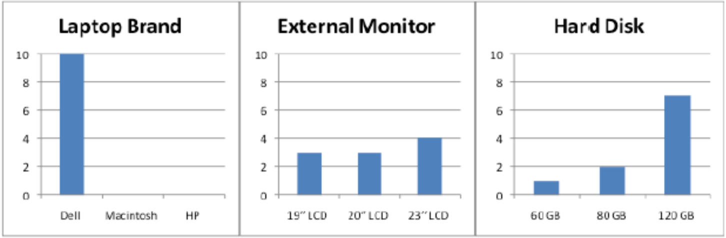
At the beginning of the negotiation, BRAM creates a bid array which contains all possible bids in the given domain. If the domain is too large to create the full bid array, a random sample of the bids is included in the array.
The bids in the array are sorted according to the utility values for BRAM.

BRAM Agent: Decision making process



BRAM Agent: Win-win offers

In order to create win-win offers, BRAM tries to create an estimation of the preferences of the other agent. The more an agent requests a certain value given an issue, the more that agent is believed to prefer that value. Since the agent may change offers over time, BRAM will only consider the last 10 bids made by the other party. BRAM creates histograms based on these final 10 bids for each domain, as can be seen below:



(a) (b) (c)

BRAM Agent: Win-win offers

As can be seen from the histograms, it is clear that the other agent clearly prefers the value Dell over Macintosh and HP in the *Laptop Brand* issue. It also seems that the other agent might be indifferent to the size of the external monitor, and it seems to prefer a 120GB Hard Disk, but the other agent might be willing to compromise on this issue.
In this example, BRAM might generate a bid with values Dell, 23" LCD and a 120GB Hard Drive. (but only if the utility of this offer is above BRAM's utility threshold)

BRAM Agent: Random offers

If BRAM fails to find a win-win offer which preserves the utility threshold, BRAM generates a random bid from the bid array that preserves the utility threshold. Since the bid array has been sorted, BRAM is able to generate a ‘nearby’ bid to the last bid it made. This ensures that the utility is comparable to its last offer. If the utility of this randomly selected offer is lower than the utility threshold, BRAM will offer the same bid as the round before it. Each offer can only be offered a set amount of times, which increases the variety of bids made by BRAM.

BRAM Agent: Utility Threshold

The utility threshold is important for BRAM to determine whether to accept an offer or to create an offer for the other agent. The utility threshold is defined as a percentage of the maximal utility that can be achieved in the current domain. This percentage decreases as time elapses, which shows us how BRAM compromises as time elapses. An example of the utility threshold used in the ANAC2011 tournament, where negotiations lasted for 180 seconds, was as follows:

- For the first 60 seconds, the utility threshold was 93% of the maximum utility
- This percentage decreased to 85% of the maximum utility during the next 90 seconds
- The 25 seconds following this, the percentage was 70% of the maximum utility
- For the final 5 seconds, the utility threshold was 20%

BRAM Agent: Conclusion

- BRAM is a cooperative agent $\Rightarrow$ good results when playing other cooperative agents
- Results were poorer when playing against non-cooperative agents due to compromising
- In many negotiations, an agreement was found at the Nash-Equilibrium
- BRAM failed to reach an agreement in only 2 of the 385 negotiations
- Because it can return another bid quickly, it is a good agent to negotiate with in a time-based environment

Table of Contents

AgentK2: Overview

AgentK2 attempts to model the utility space of the other agent by looking at the bids being made by the opponent. AgentK2 utilizes the time its been given in a negotiation. This way, AgentK2 doesn't compromise too quickly and is able too get the best chance of generating a mutual beneficial outcome.
Since it is a tournament form, AgentK2 always looks to increase its own utility as much as possible. AgentK2 is able to determine or guess the most the other agent is willing to concess, and create a bid accordingly.

AgentK2: Problems with the paper

- Advanced mathematics
- Not written in English, there just happen to be some loosely connected words in English written on the pages
- Wasn't able to find any papers on AgentK2 written in English, as opposed to this paper which has been translated by Google Translate
- We might be able to get a clearer understanding of what the authors have written, but I need help if we are going to try to decipher it
- AgentK2 is a good agent (I read that it won one of the ANAC tournaments), but after trying for more than 2 hours I still have not been able to understand what was written on the first 3 pages

AgentK2: Overview in short

- AgentK2 attempts to understand the opponents strategy. This is done by looking at what kind of offers the other agent sends. By looking at its own utilities for the received bids, AgentK2 can make an assumption what the optimal bid for the other agent is.
- AgentK2 can control how the optimal solution can be reached. If the agent is not willing to cooperate, AgentK2 utilizes a different formula to approach the bids being made by the other agent.
- The probability that AgentK2 accepts an offer increases over time
- AgentK2 will determine if a bid made by the opponent is done stochastically based on the distance to the average utility of bids.
- At the end of a negotiation, AgentK2 will change its concession degree. This is done in order to try to force the other agent to make a bid which is a bit more beneficial to AgentK2

Citations

Fishel, Radmila, Maya Bercovitch, et al. “Bram agent”. In: *Complex Automated Negotiations: Theories, Models, and Software Competitions*. Springer, 2013, pp. 213–216.

Kawaguchi, Shogo, Katsuhide Fujita, and Takayuki Ito. “Agentk2: Compromising strategy based on estimated maximum utility for automated negotiating agents”. In: *Complex Automated Negotiations: Theories, Models, and Software Competitions*. Springer, 2013, pp. 235–241.