BINGO: an Algorithm for Automated Negotiations with Hidden Reservation Values and a Fixed Number of Rounds

Dave de Jonge[a,*]

[a]IIIA-CSIC, Barcelona, Spain
ORCID (Dave de Jonge): https://orcid.org/0000-0003-2364-9497

Abstract. We present a new algorithm for automated negotiation, called BINGO, that we designed specifically for the main league of the Automated Negotiating Agents Competition of 2024 (ANL 2024). This means it was designed for negotiations that take place over a fixed number of rounds and in which the agents are fully aware of each others' utility functions, but in which their reservation values are kept private. Our algorithm is based on the principle of backward induction, combined with the assumption that the belief each agent holds about its opponent's reservation value can be modeled as a uniform probability distribution. We present an experiment in which BINGO negotiated against all finalists of ANL 2024 and we show that BINGO outperformed all of them. Furthermore, we present a number of theoretical results for so-called split-the-pie scenarios.

1 Introduction

The field of automated negotiation deals with the question of how two or more self-interested agents with conflicting goals can negotiate to find agreements that are mutually beneficial. The challenge for a negotiating agent is to find the right balance between demanding a high utility for itself on the one hand, and conceding enough to its negotiation partner to make them willing to accept the deal, on the other hand [8]. A typical example is the case where a buyer and a seller bargain over the price of a second-hand car.

To promote research on this topic, the annual Automated Negotiating Agent Competition (ANAC) has been organized since 2010 [5] and has since become the default benchmark for negotiation algorithms. Throughout the years it has dealt with various different scenarios with different characteristics. For example, it involved settings in which agents were able to learn from previous negotiation sessions [13, 24], negotiations with extremely large offer spaces [11], multilateral negotiations [12], and negotiations in which the agents only have partial knowledge about their own utility functions [2]. Furthermore, since 2017 ANAC has been extended with a number of additional 'leagues' focused on more specific challenges, such as the game of Diplomacy [10], supply chain management [20], negotiations between computers and humans [19] and the game of Werewolves [2]. Since then, the *main* league of ANAC has been referred to as the Automated Negotiations League (ANL), to distinguish it from these additional leagues.

Thus far, however, in all previous editions of ANAC it was always assumed that each agent only knew its own utility function, while the utility function of its opponent was hidden.[1] In this sense, ANL 2024 involved a major change. This time, the agents' respective utility functions were common knowledge, and instead only their so-called *reservation values* were considered private (i.e. the amount of utility each agent receives when the negotiations fail). Another major change with respect to most (but not all) previous editions of ANL, was that the agents were not only limited by a temporal deadline, but also by a maximum number of proposals [3].

Although some might argue that the these changes make the competition less realistic, they do make it more suitable for mathematical analysis, which makes it more interesting from a theoretical perspective. Indeed, it should be noted that for this reason many similar problems have been studied in the past.

For example, the problem how to find the theoretically optimal deal under full information but without any limitation on the number of proposals has been solved for various different settings [22, 26, 7]. Similarly, several authors studied negotiations under full information but with a fixed number of negotiation rounds. For example, Di Giunta and Gatti [16] used backward induction together with convex programming techniques to determine the subgame perfect equilibrium in such a scenario. Similarly, Sloof [27] studied a split-the-pie bargaining scenario in which a new pie is divided in every round and Busch and Wen [6] studied a scenario in which, after each rejection, the two agents played a one-shot 'disagreement game'.

Finally, several authors studied negotiations with a fixed number of rounds and full information about the opponent's utility functions, but with limited information about other parameters. For example, Di Giunta and Gatti [15] and Gatti *et al.* [14] studied negotiations in which one of the two agents did not know the individual deadline of its opponent. They used backward induction to find an optimal strategy and showed that (in combination with a certain belief-update rule) it forms a sequential equilibrium. Furthermore, An *et al.* [1] presented a negotiation algorithm for scenarios with private reservation values, but they assumed negotiations between a single buyer and multiple sellers, and they assumed the reservation values could only take two possible values. They did also discuss a generalization for more than two possible reservation values, but it remained unclear whether their algorithm can be implemented efficiently and they did not present any experiments.

* Corresponding Author. Email: davedejonge@iiia.csic.es

[1] Or, in the case of Diplomacy, computationally too complex to calculate.

In this paper we present a new agent, called Backward Induction NeGOtiator (BINGO) that we designed specifically for the setup of ANL 2024. To the best of our knowledge, the exact settings of ANL 2024 have never been studied before, so *the agents that participated in ANL 2024 are the state-of-the-art* and are in fact the *only* existing agents that BINGO can be compared with.

It is striking that all agents that reached the finals of this competition were based either on heuristics or on machine learning techniques.[2] In fact, the runner-up of the competition, UOAgent, even followed a purely time-based strategy. BINGO is therefore unique in the sense that it is the only agent that tries to make optimal proposals based on theoretical principles. One participant, called AntiAgent, did also use backward induction, but in addition relied on a machine learning approach to estimate the opponent's reservation value.

We have made the source code of BINGO publicly available at: https://www.iiia.csic.es/~davedejonge

2 Negotiations in ANL 2024

We here describe the setup of ANL 2024 (i.e. the main league of ANAC 2024). The negotiations in this competition can be formally modeled as a turn-taking game with imperfect information.

In this competition, each participant had to implement a negotiating agent. All submitted agents then had to negotiate against each other in a large number of different scenarios. The winner would be the one that achieved the highest average utility over all these negotiations.

Each scenario consisted of the following components (which we will explain in more detail below): two **agents**, which we denote by α_1 and α_2 respectively. A finite set of possible **offers** Ω, called the **offer space**. Two **utility functions** u_1 and u_2 for the two respective agents: $u_i : \Omega \to [0,1]$. Two **reservation values** $r_1, r_2 \in [0,1]$ for the two respective agents. A positive integer N indicating the **number of rounds** in the negotiation. And finally, a **temporal deadline** T. In each of these scenarios the two agents α_1 and α_2 were drawn from the pool of agents submitted by the participants.

2.1 Negotiation Protocol

In each scenario, the two agents had to negotiate with each other according to the *alternating offers protocol* (AOP) [25]. This means the agents had to take turns in the negotiations. Specifically, it means the negotiations proceeded as follows. In Round 1, the agent whose turn it is (say agent α_1), can pick any offer ω from the set of offers Ω and propose it to the other agent. Next, in Round 2, it is then the other agent's turn (agent α_2). She first needs to decide whether to accept or to reject the proposal she received in Round 1. If she accepts, then the negotiations are over. If she rejects, then she can make a counter-proposal. That is, α_2 can select another offer ω' from Ω and propose it to α_1. Then, in Round 3, it is α_1's turn again, so now α_1 needs to decide whether to accept or to reject the proposal ω' she received in Round 2. If she accepts, then the negotiations are over. If she rejects, then she can propose a new offer to α_2. Etcetera. The negotiations can end in either of the following three ways: 1) Either of the two agents accepts a proposal. 2) After N rounds neither of the two agents has accepted any proposal. 3) The negotiations have lasted more than T seconds. In the first case, the agents α_1 and α_2 receive the respective utility values $u_1(\omega)$ and $u_2(\omega)$ corresponding to the accepted offer ω and their respective utility functions u_1 and u_2. In this case we say the accepted offer ω has become an **agreement**. In the other two cases, however, the negotiations end without agreement, and the respective utility values received by the agents equal their respective reservation values r_1 and r_2.

For each agent submitted to the competition, its score was calculated by taking the average over all negotiations in which it participated, of the utility it received minus its reservation value. In each scenario of ANL 2024, the size of the offer space was between 900 and 1100 offers, and the number of rounds N varied between 10 and 10,000. The temporal deadline was always set to $T = 180$ seconds.

In the rest of this paper, for any given round n, we refer to the agent whose turn it is in that round as the **active agent** of round n and we refer to the other agent as the **passive agent** of round n. Furthermore, we will refer to α_2 as the **opponent** of α_1, and vice versa. We may use the notation α_i as a variable to refer to either of the two agents, and in that case the notation α_{-i} refers to the opponent of α_i. Similarly, u_i and r_i refer to the utility function and reservation value of α_i, and u_{-i} and r_{-i} are the corresponding quantities for α_{-i}.

2.2 Reservation Values

The importance of the reservation values, is that for each agent its reservation value is the minimum utility she is already guaranteed to receive, even without coming to an agreement. Therefore, a rational agent would never accept any offer that yields less utility than her reservation value.

In ANL 2024 each agent had full knowledge of the negotiation scenario, except for the opponent's reservation value. That is, each agent α_i knew its own utility function u_i and reservation value r_i, as well as the opponent's utility function u_{-i}, but not the opponent's reservation value r_{-i}. However, the agents did have access to the knowledge that each agent's reservation value r_i was drawn from a uniform probability distribution[3] over the interval $[0, b_i^1]$ with:

$$b_i^1 = u_i(\omega^{pm}) - \epsilon \tag{1}$$

where ϵ is a small constant and ω^{pm} is the 'product maximizing' offer: $\omega^{pm} := \arg\max_{\omega \in \Omega} u_1(\omega) \cdot u_2(\omega)$. The reason for the superscript 1 in the notation b_i^1 will become clear later.

3 Belief Update

As explained, the agents in ANL 2024 did not know each other's reservation values. However, the idea behind BINGO is that we assume that each agent does have a *belief* about its opponent's reservation value, which we model as a probability distribution. That is, let $P_i^n(x)$ denote the probability density that α_{-i} assigns, in round n of the negotiations, to the the hypothesis that α_i's reservation value equals x (i.e. the probability that $r_i = x$). Furthermore, we assume it is common knowledge among the agents that each reservation value r_i was drawn uniformly from an interval $[0, b_i^1]$, with b_i^1 as defined by Eq. (1). So, for the *initial* beliefs P_i^1, we have:

$$P_i^1(x) = \begin{cases} \frac{1}{b_i^1} & \text{if } x \in [0, b_i^1] \\ 0 & \text{otherwise} \end{cases}$$

These beliefs may be updated throughout the negotiations, based on the proposals the agents receive from each other.

[2] We know this because every participant wrote a short report detailing their strategy.

[3] While this was not explicitly communicated to the participants, it could be found in the source code of the tournament software which was available to the participants.

In addition, we assume that the agents are perfectly rational, and that each agent assumes its opponent is perfectly rational. This implies that whenever an agent α_i proposes an offer ω, its opponent will know that α_i's reservation value r_i must be below α_i's utility value for ω. That is: $r_i < u_i(\omega)$. The agents may therefore use this information to update their beliefs. There are many ways how one could do this, but for our implementation of BINGO we assumed that the belief P_i^n always remains a uniform distribution, and that only the upper bound of its support (which we will denote by b_i^n) is adapted to equal the minimum utility demanded by the opponent so far. That is, at any round n of the negotiation, α_{-i}'s belief is given by:

$$P_i^n(x) = \begin{cases} \frac{1}{b_i^n} & \text{if } x \in [0, b_i^n] \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where b_i^n is given by:

$$b_i^n := \min \ \{u_i(\omega) \mid \omega \text{ has been proposed by } \alpha_i\} \cup \{b_i^1\} \tag{3}$$

We should stress that BINGO is based on the assumption that *both* agents update their beliefs according to Eqs. (2) and (3), and therefore that the values b_1^n and b_2^n are *common knowledge*. In other words, each agent α_i knows the belief that the other agent α_{-i} holds about α_i's reservation value r_i. Of course, this assumption will typically not hold, unless BINGO negotiates against itself, but we will see later that our experiments show BINGO still works very well, even when it negotiates against other agents.

4 Backward Induction

We are now ready to present BINGO. It uses a well-known technique from game theory, called backward induction [23], to determine for each round of the negotiation whether or not the active agent should accept the previous proposal and, if not, which offer it should then propose. Specifically, it starts by determining the best action for the last round (round N), and then, knowing this, it works its way backwards, by next determining the best action for round $N-1$, then the best action for round $N-2$, etcetera.

In the rest of this paper we will assume, w.l.o.g. that agent α_2 is the active agent in round N. This implies that α_2 is also the active agent is any round $N-j$ where j is an even number and that α_1 is the active agent in any round $N-j$ where j is an odd number.

We will use the predicate $\mathfrak{acc}(i, \omega, n)$, with $i \in \{1, 2\}, \omega \in \Omega$, and $n \in \{1, 2, \ldots, N\}$, to denote that agent α_i would accept an offer ω in round n. Alternatively, we may also just write "α_i accepts" if the offer and round are clear from context. Furthermore, we may write "α_i rejects" to indicate that α_i would not accept the offer. We will use the notation ω^n to denote the offer proposed in round n.

Furthermore, for any offer ω, round n, and agent α_i we will define the notion of the **anticipated utility**, which is the utility that agent α_i can expect to receive *at the end of the negotiations*, given that she proposes or receives the offer ω in round n. We use the notation U_a^n to denote the anticipated utility for the *active* agent in round n and U_p^n to denote the anticipated utility for the *passive* agent in round n. We may use the term **direct utility** to refer to the ordinary utility functions u_i that are given with the negotiation scenario, to stress that we are not talking about *anticipated* utility.

The idea is that BINGO always proposes the offer that maximizes the anticipated utility, unless the direct utility of the offer it just received is even higher (or equal), in which case BINGO would accept that offer.

4.1 Round N

The optimal strategy for α_2 in the last round is simple: accept the last offer proposed by the opponent (denoted ω^{N-1}) if and only if its direct utility lies above α_2's reservation value:

$$\mathfrak{acc}(2, \omega^{N-1}, N) \quad \leftrightarrow \quad r_2 < u_2(\omega^{N-1}) \tag{4}$$

It is important to note here that ANL 2024 did not allow agents to learn or collect data between negotiation sessions. Each negotiation was entirely independent from every other negotiation. This is important because otherwise it could sometimes be better for α_2 to reject offer ω^{N-1} even if its utility is strictly above r_2, since this could force the opponent to concede more in future negotiations.

Note that in round N, if α_2 does not accept the proposal, then it is not necessary to determine which offer to propose instead, because the negotiations are already finished anyway.

4.2 Round $N-1$, agent α_1

In the second-last round of the negotiations (round $N-1$), agent α_1 is the active agent. To determine the best offer for α_1 to propose in this round we need to calculate the anticipated utility for all offers $\omega \in \Omega$, and then select the offer that maximizes this value.

Let ω be any arbitrary offer. Suppose that in round $N-1$ agent α_1 proposes this offer ω. This means that in the final round the other agent α_2 will accept this proposal iff $r_2 < u_2(\omega)$, while the negotiations will end without agreement otherwise. Therefore, given the reservation values r_1, r_2 we can, for any offer ω, define the anticipated utility for α_1 as:

$$U_a^{N-1}(\omega, r_1, r_2) = \begin{cases} u_1(\omega) & \text{if } r_2 < u_2(\omega) \\ r_1 & \text{otherwise} \end{cases} \tag{5}$$

However, α_1 cannot use this directly to select the best offer, because it requires knowledge of the opponent's reservation value r_2. Therefore, we first need to calculate the *expected* anticipated utility EU_a^{N-1}, by integrating over all possible values of r_2:

$$EU_a^{N-1}(\omega, r_1, b_2^{N-1}) = \int_0^1 P_2^{N-1}(r_2) \cdot U_a^{N-1}(\omega, r_1, r_2) \cdot dr_2 \tag{6}$$

Recall from Section 3 that α_1's belief about r_2 in round $N-1$, is given by a uniform probability distribution P_2^{N-1} over the interval $[0, b_2^{N-1}]$. To simplify notation we will below just write b_2 instead of b_2^{N-1}.

To calculate this integral, we have to distinguish between the following two cases:

$$u_2(\omega) \in [0, b_2] \quad \text{and} \quad u_2(\omega) \in [b_2, 1]$$

In the first case the expected anticipated utility can be calculated by plugging Eqs. (2) and (5) into Eq. (6), to get:

$$\begin{aligned} EU_a^{N-1}(\omega, r_1, b_2) &= \frac{1}{b_2} \int_0^{b_2} U_a^{N-1}(\omega, r_1, r_2) \cdot dr_2 \\ &= \frac{1}{b_2} \int_0^{u_2(\omega)} u_1(\omega) \cdot dr_2 + \frac{1}{b_2} \int_{u_2(\omega)}^{b_2} r_1 \cdot dr_2 \\ &= \frac{1}{b_2} \cdot u_2(\omega) \cdot (u_1(\omega) - r_1) + r_1 \end{aligned}$$

On the other hand, in the second case we know that $u_2(\omega)$ is certainly greater than r_2, and we have seen in Section 4.1 that this means that

α_2 will certainly accept the offer, so α_1 will certainly receive the utility value $u_1(\omega)$. So, in that case we have:

$$EU_a^{N-1}(\omega, r_1, b_2) \quad = \quad u_1(\omega)$$

Combining the two cases we have:

$$EU_a^{N-1}(\omega, r_1, b_2) = \begin{cases} \frac{1}{b_2} \cdot u_2(\omega) \cdot (u_1(\omega) - r_1) + r_1 & \text{if } u_2(\omega) \leq b_2 \\ u_1(\omega) & \text{otherwise} \end{cases} \tag{7}$$

Now, given α_1's reservation value r_1 and an upper bound b_2 for α_2's reservation value, we can find the optimal offer ω^{*N-1} for α_1 to propose in round $N-1$ by maximizing over the expected anticipated utility:

$$\omega^{*N-1} \;:=\; \mathcal{O}^{N-1}(r_1, b_2) \;:=\; \arg\max_{\omega \in \Omega} \; EU_a^{N-1}(\omega, r_1, b_2) \tag{8}$$

Here, we use the notation $\mathcal{O}^{N-1}$ to denote the *function* that calculates the optimal offer, while we use ω^{*N-1} to denote the optimal offer itself. That is, ω^{*N-1} is the *output* of the function $\mathcal{O}^{N-1}$.

However, before proposing this offer, α_1 should first decide whether or not to accept the offer ω^{N-2} it received in the previous round. BINGO will accept if and only if the direct utility of ω^{N-2} is greater than the expected anticipated utility of the offer ω^{*N-1} it would otherwise propose:

$$\begin{aligned} \mathfrak{acc}(1, \omega^{N-2}, N-1) &\leftrightarrow EU_a^{N-1}(\omega^{*N-1}, r_1, b_2) < u_1(\omega^{N-2}) \\ &\leftrightarrow \max_{\omega \in \Omega} \; EU_a^{N-1}(\omega, r_1, b_2) < u_1(\omega^{N-2}) \end{aligned}$$

4.3 *Round* $N-1$*, agent* α_2

Let us now calculate the anticipated utility U_p^{N-1} that α_2 assigns to the situation that α_1 proposes its optimal offer ω^{*N-1} in round $N-1$. This quantity will turn out important in Section 4.4.

If α_1 indeed proposes ω^{*N-1}, then α_2 will receive $u_2(\omega^{*N-1})$ if she accepts that offer in round N, and r_2 if she rejects it. Thus:

$$U_p^{N-1}(r_1, r_2, b_2) = \begin{cases} u_2(\omega^{*N-1}) & \text{if } \alpha_2 \text{ accepts} \\ r_2 & \text{if } \alpha_2 \text{ rejects} \end{cases}$$

Since we know from Section 4.1 that α_2 accepts it if and only if $r_2 < u_2(\omega^{*N-1})$, this becomes:

$$U_p^{N-1}(r_1, r_2, b_2) = \begin{cases} u_2(\omega^{*N-1}) & \text{if } r_2 < u_2(\omega^{*N-1}) \\ r_2 & \text{otherwise} \end{cases} \tag{9}$$

which can also be written as:

$$U_p^{N-1}(r_1, r_2, b_2) \;=\; \max \; \{ \; u_2(\omega^{*N-1}) \;,\; r_2 \; \}$$

Note that while the quantities r_1 and b_2 do not appear on the right-hand side explicitly, the expression does depend on them implicitly because ω^{*N-1} depends on them through Eq. (8).

4.4 *Round* $N-2$

Now, let us show how to determine the optimal offer ω^{*N-2} for α_2 to propose in turn $N-2$, assuming that in round $N-1$ the opponent α_1 will follow the strategy given in Section 4.2.

We follow the same recipe as above. That is, we first find an expression for the anticipated utility U_a^{N-2} of α_2, for any arbitrary offer ω and any arbitrary reservation values. We then integrate over all possible values of r_1, and finally find the offer that maximizes this quantity.

Again, let ω be any arbitrary offer. If, in round $N-2$, agent α_2 proposes ω and in round $N-1$ agent α_1 accepts it, then α_2 will receive its corresponding utility value $u_2(\omega)$. On the other hand, if α_1 rejects it, then α_1 will propose her optimal offer ω^{*N-1}, so α_2 will receive the expected anticipated utility corresponding to that scenario, which is given in Section 4.3. Thus:

$$U_a^{N-2}(\omega, r_1, r_2, b_2^{N-2}) = \begin{cases} u_2(\omega) & \text{if } \alpha_1 \text{ accepts.} \\ U_p^{N-1}(r_1, r_2, b_2^{N-1}) & \text{if } \alpha_1 \text{ rejects.} \end{cases} \tag{10}$$

It is important to note here, that b_2^{N-2} represents the belief that α_1 holds in round $N-2$, about r_2, while b_2^{N-1} represents α_1's belief in round $N-1$, which is after α_2 has proposed the offer ω. This means that, according to to Eq. (3) these two quantities are related by the following equation:

$$b_2^{N-1} \quad = \quad \min \; \{ \; b_2^{N-2} \;,\; u_2(\omega) \; \}$$

Of course, α_1 will only accept an offer ω if it yields at least as much as what she expects to receive if negotiations continue, which we calculated in Section 4.2. So we have:

$$\alpha_1 \text{ accepts } \;\leftrightarrow\; EU_a^{N-1}(\omega^{*N-1}, r_1, b_2^{N-1}) \leq u_1(\omega) \tag{11}$$

Similar as above, we can then calculate the expected anticipated utility of α_2 (we use b_1 and b_2 as shorthands for b_1^{N-2} and b_2^{N-2}):

$$EU_a^{N-2}(\omega, r_2, b_1, b_2) = \frac{1}{b_1} \int_0^{b_1} U_a^{N-2}(\omega, r_1, r_2, b_2) \cdot dr_1$$

And finally, the optimal offer ω^{*N-2} is then determined as the one that maximizes the expected anticipated utility of α_2:

$$\omega^{*N-2} := \mathcal{O}^{N-2}(r_2, b_1, b_2) := \arg\max_{\omega \in \Omega} EU_a^{N-2}(\omega, r_2, b_1, b_2)$$

4.5 *Round* n

We will now generalize the previous sections to arbitrary rounds. That is, we will show how, for any $n \in \{1, 2, \ldots, N\}$, we can determine the optimal offer ω^{*n} to be proposed by the active agent of round n.

In the previous sections we have introduced the following operators: U_a^n, EU_a^n, U_p^n, and $\mathcal{O}^n$, which each depend on (some of) the following five parameters: ω, r_1, r_2, b_1^n, and b_2^n. We will first define these operators for $n = N$ and then provide the recursion relations that allows us to calculate them for any n with $n < N$.

Note that U_a^N is defined as the utility that α_2 would receive when she proposes some given offer ω in the last round. However, since it is the last round, this offer cannot get accepted, and therefore the utility that α_2 receives is her reservation value. We therefore have:[4] $U_a^N(\omega, r_1, r_2, b_1^N, b_2^N) = r_2$ for any offer ω. For the same reason, her *expected* utility will also be her reservation value: $EU_a^N(\omega, r_1, b_1^N, b_2^N) = r_2$. And similarly, for agent α_1 the utility

[4] Note that we include all parameters on which U_a^n depends, for general n, even though for the particular case of $n = N$ it only depends on r_2. In this subsection we do the same for the other operators.

she receives whenever α_2 does not accept in round N is given by: $U_p^N(r_1, r_2, b_1^N, b_2^N) = r_1$.

Given these initial equations, we can now recursively calculate U_a^n, EU_a^n, U_p^n, and $\mathcal{O}^n$ for all rounds n as follows (here, α_i always denotes the active agent of round n, and α_{-i} the passive agent of round n):

$$U_a^n(\omega, r_1, r_2, b_1^n, b_2^n) \quad := \quad \begin{cases} u_i(\omega) & \text{if } \mathfrak{acc}(-i, \omega, n+1) \\ U_p^{n+1}(r_1, r_2, b_1^{n+1}, b_2^{n+1}) & \text{otherwise} \end{cases}$$

where ω can be any arbitrary offer and where, by Eq. (3):

$$b_i^{n+1} := \min\{b_i^n, u_i(\omega)\} \quad \text{and} \quad b_{-i}^{n+1} := b_{-i}^n$$

Furthermore, note that $\mathfrak{acc}(-i, \omega, n+1)$ and U_p^{n+1} are determined by the equations below.

$$EU_a^n(\omega, r_i, b_1^n, b_2^n) \;:=\; \frac{1}{b_{-i}^n} \int_0^{b_{-i}^n} U_a^n(\omega, r_1, r_2, b_1^n, b_2^n)\, dr_{-i}$$

where ω can be any arbitrary offer.

$$\omega^{*n} \;:=\; \mathcal{O}^n(r_i, b_1^n, b_2^n) \;:=\; \arg\max_{\omega \in \Omega} EU_a^n(\omega, r_i, b_1^n, b_2^n)$$

$$U_p^n(r_1, r_2, b_1^n, b_2^n) := \begin{cases} u_{-i}(\omega^{*n}) & \text{if } \mathfrak{acc}(-i, \omega^{*n}, n+1) \\ EU_a^{n+1}(\omega^{*n+1}, r_i, b_1^{n+1}, b_2^{n+1}) & \text{otherwise} \end{cases}$$

where $b_i^{n+1} := \min\{b_i^n, u_i(\omega^{*n})\}$ and $b_{-i}^{n+1} := b_{-i}^n$.

Finally, in any round n with $n > 1$, BINGO will accept the offer ω^{n-1} it received in round $n-1$, if and only if its utility is greater than or equal to the expected anticipated utility associated with the optimal offer ω^{*n} that BINGO would otherwise propose. That is:

$$\mathfrak{acc}(i, \omega, n) \quad \leftrightarrow \quad EU_a^n(\omega^{*n}, r_i, b_1^n, b_2^n) < u_i(\omega)$$

It is interesting to note that, despite the fact that an agent α_i exactly knows her own reservation value r_i, her optimal offers do not only depend on r_i itself, but *also* on the upper bound b_i for her own reservation value. This is because α_i's optimal proposal depends on which actions her opponent α_{-i} is going to take in the coming rounds, which in turn depend on α_{-i}'s belief about α_i's reservation value.

5 Theoretical Results

In this section we will present a number of theoretical results related to BINGO and the scenarios of ANL 2024.

The fact that the agents do not know each others' reservation values means that the agents are playing a game of *incomplete information*. The appropriate equilibrium concept for such games is known as *sequential equilibrium* [18]. This means there is not only an equilibrium among the agents' strategies, but also between their respective *beliefs* about each other's types. Unfortunately, however, it is well-known that calculating such equilibria is a very hard problem and existing approaches to find such equilibria only work for simple toy-world games [17]. It would therefore seem intractable to find such equilibria for the ANL 2024 scenarios, and so we leave it as an open conjecture that BINGO forms a sequential equilibrium.

Conjecture 1. *When the two agents both apply BINGO their joint strategies and beliefs form a sequential equilibrium.*

However, if we ignore the agents' freedom to choose their own belief update rules and instead assume that they would all use the same belief update rules as BINGO, then the game reduces to an ordinary game of full information, so we can obtain the following result.

Theorem 1. *If we make the assumption that all agents follow the beliefs given by Eqs. (1)-(3) then BINGO vs. BINGO forms a subgame perfect equilibrium.*

Proof. Under the given assumption, the agents' uncertainties about their respective reservation values become common knowledge and therefore the uncertainty about whether or not any given offer will be accepted can be modeled as an ordinary random variable with a commonly known probability distribution. Under this model, the negotiations become a non-deterministic game of *full information* and for such games the subgame perfect equilibrium can be found by backward induction [23], which is exactly what BINGO does. ☐

In general, the equations from Section 4 are too difficult to solve analytically, especially since the utility functions u_i may take many different forms. However, we will show that for a simple type of scenario known as a 'split-the-pie' scenario, at least for rounds $N-1$ and $N-2$ we can find explicit expressions for the optimal offers to propose.

Definition 1. *A* ***split-the-pie scenario*** *is a negotiation scenario for which we have $u_1(\omega) + u_2(\omega) = 1$, for every offer $\omega \in \Omega$. Furthermore, we say it is an* ***idealized split-the-pie scenario*** *if, in addition, we have that for any $x \in [0,1]$ there exists some $\omega \in \Omega$ such that $u_1(\omega) = x$.*

Note that around 25% of the scenarios used in the ANL 2024 competition were split-the-pie scenarios. Of course, these scenarios were non-idealized, because such scenarios would have an infinite number of offers.

For the theorems below to hold exactly, we need to make the further assumption that $\epsilon \leq \frac{1}{2}r_1$ in Eq. (1), which is indeed very likely to be true, since ϵ is sure to be very small.

Theorem 2. *For any idealized split-the-pie scenario, the optimal offer ω^{*N-1} for agent α_1 to propose in round $N-1$, is the one for which $u_1(\omega^{*N-1}) = \frac{1}{2} + \frac{1}{2}r_1$, unless α_2 has earlier already proposed one or more offers with higher utility for α_1, in which case α_1 should just accept or re-propose the best such offer.*

Note that a similar result was also found in [4], but under different assumptions.

Proof. Starting from Eq. (7) and using the fact that in split-the-pie scenarios we have $u_2 = 1 - u_1$, we get the following expression:

$$EU_a^{N-1}(\omega, r_1, b_2) = \begin{cases} \frac{1}{b_2} \cdot (1 - u_1(\omega)) \cdot (u_1(\omega) - r_1) + r_1 & \text{if } 1 - b_2 \leq u_1(\omega) \\ u_1(\omega) & \text{otherwise} \end{cases} \tag{12}$$

It is then easy to show, using straightforward algebra (see Appendix A of the supplementary material of this paper [9]), that the offer ω^{*N-1} that maximizes this quantity is the one that satisfies:

$$u_1(\omega^{*N-1}) \quad = \quad \max\left\{ \frac{1}{2} + \frac{1}{2}r_1 \;,\; 1 - b_2 \right\} \tag{13}$$

To interpret this equation, let us define ω_{min} to be the offer with lowest utility for α_2 that has so far already been proposed by α_2:

$$\omega_{min} := \arg\min \ \{u_2(\omega) \mid \omega \text{ has been proposed by } \alpha_2\} \quad (14)$$

Now, recall that b_2 is shorthand for b_2^{N-1}, so by Eq. (3) we have $b_2 = \min\{u_2(\omega_{min}), b_2^1\}$. Furthermore, note that for idealized split-the-pie scenarios we have $b_2^1 = \frac{1}{2} - \epsilon$ (by Eq. (1)), so we have $b_2 = \min\{u_2(\omega_{min}) \ , \frac{1}{2} - \epsilon\}$ which means $1 - b_2 = \max\{1 - u_2(\omega_{min}) \ , \frac{1}{2} + \epsilon\}$ and because we are talking about a split-the-pie scenario: $1 - b_2 = \max\{u_1(\omega_{min}) \ , \frac{1}{2} + \epsilon\}$. Combining this with Eq. (13) we get:

$$\begin{aligned} u_1(\omega^{*N-1}) &= \max \ \{ \ \frac{1}{2} + \frac{1}{2}r_1 \ , \ u_1(\omega_{min}) \ , \ \frac{1}{2} + \epsilon \ \} \\ &= \max \ \{ \ \frac{1}{2} + \frac{1}{2}r_1 \ , \ u_1(\omega_{min}) \ \} \end{aligned}$$

Here we used our assumption that $\epsilon \leq \frac{1}{2}r_1$, but it is easy to see that without this assumption, the result would still be approximately correct, as long as ϵ is very small.

So, we have shown that α_1 will propose an offer with utility $\frac{1}{2} + \frac{1}{2}r_1$, unless the offer ω_{min} that she has already received from α_2 is actually better for her, in which case ω_{min} is the optimal offer. Furthermore, combining the definition of ω_{min} (Eq. (14)) with the fact that we are talking about a split-the-pie scenario, we get:

$$\omega_{min} = \arg\max \ \{u_1(\omega) \mid \omega \text{ has been proposed by } \alpha_2\}$$

which means that, from the point of view of α_1, offer ω_{min} is indeed the *best* offer she has received. □

Theorem 3. *In a split-the-pie scenario, in round $N-2$, any offer that satisfies $u_2(\omega) > \max\{r_2, \frac{1}{2}\}$ is optimal.*

The proof can be found in Appendix B of the supplementary material of this paper [9].

6 Implementation

We here discuss the implementation of BINGO. We should stress, however, that we here just present a naïve version of the implementation, while the real implementation involves a number of optimizations to make the code more efficient. We discuss some of these optimizations at the end of this section. We have implemented BINGO in Python, on the NegMas platform [21], which is the platform on which the ANL 2024 competition was run.

6.1 Calculations

Since the equations in Section 4 are generally too hard to solve analytically, BINGO instead solves them numerically. We will see that the accuracy of these calculations depends on a parameter K. The higher its value, the more accurate the calculations, but the slower the algorithm will be and the more memory it will need. So, we had to select the highest value of K that still allowed us to run our experiments on our hardware, within the time limits of the competition. After some trial-and-error we found this value to be $K = 50$ (but this of course this number can be different if one uses different hardware).

Given this parameter we define:

$$R_K := \{0 \ , \ \frac{1}{K} \ , \ \frac{2}{K} \ , \ \ldots \ , \ \frac{K-1}{K} \ , \ 1\}.$$

At the beginning of the negotiations, BINGO starts by creating a 3-dimensional array of size $|\Omega| \times (K+1)^2$ and filling it with the values of $U_a^{N-1}(\omega, r_1, r_2)$, as given by Eq. (5), for all offers $\omega \in \Omega$ and all possible values of r_1, and r_2 in R_K.

Next, BINGO creates another 3-dimensional array to store the values of $EU_a^{N-1}(\omega, r_1, b_2^{N-1})$ for all offers $\omega \in \Omega$ and all possible values of r_1 and b_2^{N-1} in R_K. Instead of performing the actual integral of Eq. (6), the values of EU_a^{N-1} are approximated by summing $U_a^{N-1}(\omega, r_1, r_2)$ over the possible values of r_2 in R_K, between 0 and b_2^{N-1}. That is:

$$EU_a^{N-1}(\omega, r_1, b_2^{N-1}) \approx \frac{1}{k'} \sum_{k=0}^{k'} U_a^{N-1}(\omega, r_1, \frac{k}{K}) \quad (15)$$

where k' is the largest integer satisfying $\frac{k'}{K} \leq b_2^{N-1}$.

Then, BINGO creates a 2-dimensional array to store the optimal offers $\mathcal{O}^{N-1}(r_1, b_2^{N-1})$ for each possible value of r_1 and b_2^{N-1} in R_K, calculated according to Eq. (8). Finally, BINGO creates another 3-dimensional array to store the values of U_p^{N-1}, for all possible values of r_1, r_2, and b_2^{N-1} in R_K, according to Eq. (9).

This can then, in principle, be repeated for rounds $N-2$, $N-3$, etcetera, using the equations of Section 4.5. However, we have configured BINGO to stop doing this once it has finished its calculations for round $N-3$, because it would take too much time and memory to continue any further.

Note that if BINGO is α_1 then it knows the exact value of its own reservation value r_1. Nevertheless, it still needs to calculate $\mathcal{O}^{N-1}(r_1, b_2^{N-1})$ for *all* possible values of r_1 in R_K. This is because these values are required for the calculation of EU_a^{N-2}, which is calculated from the point of view of agent α_2, which does not know r_1. And the values of EU_a^{N-2} are in turn necessary for BINGO to calculate its optimal offer during round $N-3$.

In order to check that BINGO was implemented correctly, we verified that, in split-the-pie scenarios, the optimal offers ω^{*N-1} and ω^{*N-2} calculated by BINGO were indeed very close to the theoretically optimal offers as determined in Section 5. Specifically, we observed that the relative difference between the calculated values and the theoretically predicted values was almost never greater than 2%. The fact that these values are not perfectly equal can be attributed to the fact that the split-the-pie scenarios were non-idealized, plus the fact that the accuracy of the algorithm is limited by the parameter K.

6.2 Negotiation Strategy

The calculation mentioned above are all performed during the early rounds of the negotiations. Meanwhile, BINGO will simply propose its most selfish offer (the one with highest direct utility) in each round, until the negotiations reach round $N-3$. From that moment onward, in each round n in which BINGO is the active agent, it will pick an offer to propose from the array representing $\mathcal{O}^n$. This array contains optimal offers for all possible values of r_1 and b_2^n, but BINGO picks the one corresponding to its actual reservation value r_1 and the actual value of b_2^n at that moment. BINGO will then propose that offer, as long as its anticipated utility is greater than the direct utility of the offer it received in the previous round. If this is not the case, then BINGO will accept the received offer.

6.3 Optimizations

The main bottleneck for BINGO, is the large number of values it needs to calculate. For each round of the negotiations, BINGO calculates the values of the four operators U_a^n, EU_a^n, U_p^n, and $\mathcal{O}^n$ for all possible values of their parameters. Luckily, however, we can make use of a number of optimizations that reduce the number of calculations and the sizes of the arrays.

For example, we do not actually need to store the values of U_a^n in an array, because each of its values is used only once (in the calculation of EU_a^n), so they can be discarded immediately after being calculated. Secondly, instead of doing the above calculations for all offers $\omega \in \Omega$, we only need to do them for all *Pareto-optimal* offers. Furthermore, note that if BINGO plays the role of α_2 (i.e. it is the active agent in rounds N and $N-2$) then it will never use the optimal offers calculated for round $N-3$, so BINGO can already stop its calculations after round $N-2$. Finally, we do not need to calculate EU_a^{N-3}, U_p^{N-3}, or $\mathcal{O}^{N-3}$ for *all* possible values of r_1, because, as explained at the end of Section 6.1, this would only be necessary if we wanted to calculate EU_a^{N-4}. So, if we stop at $N-3$, then α_1 only needs to calculate them for its *actual* reservation value r_1, and α_2 does not need to calculate them at all.

Each of these optimizations were indeed implemented by us in the agent that we used for our experiments.

6.4 Complexity

Note that for arbitrary n, the array representing EU_a^n may have size $|\Omega| \times (K+1)^4$, the array for U_p^n may have size $(K+1)^4$ and the array for $\mathcal{O}^n$ may have size $(K+1)^3$. However, using the above optimizations, plus the fact that BINGO does not calculate these arrays for rounds earlier than $N-3$, and the fact that for some rounds $n \geq N-3$ these operators do not depend on all parameters (e.g. U^n in general depends on all 5 parameters, but U^{N-1} only depends on ω, r_1 and r_2, as we see from Eq. (5)), we note that the largest array is EU_a^{N-3} with size $|\hat{\Omega}| \times (K+1)^3$, where $\hat{\Omega}$ is the set of Pareto-optimal offers. So, we conclude:

Observation 1. *BINGO has a worst-case space-complexity of* $O(|\hat{\Omega}| \times K^3)$.

Regarding the time-complexity, note that calculating one value of U_a^n or U_p^n is easy, because it just involves the comparison of two previously calculated numbers (see Section 4.5). So, the time it takes to fill the entire array representing U_p^n is proportional to its size, which, with the above optimizations, is in the worst case $(K+1)^3$. To calculate one entry of the array representing $\mathcal{O}^n$ we need to maximize over all $\omega \in \hat{\Omega}$, while it has $(K+1)^3$ entries. Calculating the entire array therefore has a time-complexity of $O(|\hat{\Omega}| \times K^3)$. Finally, we see from Eq. (15) that to calculate one value of EU_a^n, we need to calculate a sum with at most K values. Since the array representing EU_a^n has at most $|\hat{\Omega}| \times (K+1)^3$ entries, the time-complexity of this step is $O(|\hat{\Omega}| \times K^4)$.

Observation 2. *BINGO has a worst-case time-complexity of* $O(|\hat{\Omega}| \times K^4)$.

7 Experiments

Unfortunately, BINGO was developed after the ANL 2024 competition had already finished, so BINGO could not participate. However, we managed to simulate a competition that did include BINGO, by letting BINGO negotiate against the 10 finalists of ANL 2024, as well as against itself, under the same conditions as the competition, and then recalculating the scores of all 11 agents by averaging over all results of our own experiments plus the results of the actual competition itself.[5]

[5] These results were made available to us by the organizers.

Recall that the finalists of ANL 2024 are the state-of-the-art and the only existing algorithms that BINGO can be compared to.

In the ANL 2024 finals each agent negotiated 1952 times with every other agent, plus itself. So, in our experiments we did the same and let BINGO play 1952 negotiations against every ANL finalist (i.e. $10 \times 1952 = 19520$ negotiations) plus 1952 more negotiations against itself. For our experiments we let the NegMas platform randomly generate an entirely new negotiation scenario for every single negotiation. We used the default settings for ANL 2024, which ensures that scenarios of various different types are created (e.g. split-the-pie domains). The experiments were performed on a laptop with 13th Gen. Intel Core i7-13700H 2.40 GHz CPU and 32 GB RAM.

The results of this experiment are displayed in Table 1 (we multiplied the scores by 1000, for the purpose of readability). We see that BINGO clearly outperformed all other agents. We have performed a Welch t-test to confirm that the difference between BINGO and the number two, Shochan, was statistically significant with $p < 10^{-11}$. We also performed an empirical game theoretical evaluation, which suggested that, among these agents, BINGO vs. BINGO was the only pure empirical Nash equilibrium. However, the difference between the scores of the agents was not large enough to draw this conclusion with sufficient statistical confidence.

Rank	Agent	Score	± std. err.
1	BINGO	419.6	± 2.2
2	Shochan	399.7	± 1.9
3	UOAgent	393.7	± 1.7
4	AgentRenting2024	387.7	± 1.8
5	AntiAgent	383.4	± 2.0
6	HardChaosNegotiator	335.3	± 1.9
7	KosAgent	332.2	± 1.7
8	Nayesian2	313.4	± 1.6
9	CARCAgent	304.7	± 1.6
10	BidBot	253.6	± 1.5
11	AgentNyan	249.7	± 1.8

Table 1. Outcome of a tournament between all finalists of ANAC 2024, plus BINGO.

8 Discussion

The main open question is to what extent the belief-update rule defined by Eqs. (2) and (3) is realistic. The problem with this model is that if, in some round n, agent α_1 calculates its optimal offer ω^{*n} as a function of its reservation value r_1, then it might be possible for α_2 to invert this calculation and deduce the exact value of r_1 from ω^{*n}. In fact, we know that for split-the-pie scenarios this is indeed possible in round $N-1$, because BINGO would propose an offer with utility $\frac{1}{2} + \frac{1}{2}r_1$, from which α_2 can easily deduce the value of r_1. In that case, however, it does not matter because in round N agent α_2 cannot use that information anymore anyway. Furthermore, according to Theorem 3, in round $N-2$, agent α_1 cannot deduce any information from the optimal offer at all, beyond what is already encoded in our belief-update rule. However, for more general scenarios, it remains an open question to what extent the strategy employed by BINGO leaks information about its reservation value, and to what extent such information can be exploited.

Acknowledgements

This work was supported by a Ramón y Cajal research grant from the Spanish Ministry of Science and Innovation (RYC2022-035229-I) and by Grant no. TED2021-131295B-C31 funded by MCIN/AEI

/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR.

We thank ANAC organizers Tim Baarslag and Tamara Florijn for their help setting up the experiments.

References

[1] B. An, N. Gatti, and V. R. Lesser. Alternating-offers bargaining in one-to-many and many-to-many settings. *Ann. Math. Artif. Intell.*, 77(1-2): 67–103, 2016. doi: 10.1007/S10472-016-9506-X. URL https://doi.org/10.1007/s10472-016-9506-x.

[2] R. Aydogan, T. Baarslag, K. Fujita, J. Mell, J. Gratch, D. de Jonge, Y. Mohammad, S. Nakadai, S. Morinaga, H. Osawa, C. Aranha, and C. Jonker. Research challenges for the automated negotiating agents competition (anac) 2019. In *Multi-Agent Systems and Agreement Technologies. 17th International Conference EUMAS 2020 and 7th International Conference AT 2020. Thessaloniki, Greece September 14-15, 2020. Revised Selected Papers*. Springer, 2020.

[3] R. Aydogan, T. Baarslag, T. C. P. Florijn, K. Fujita, C. M. Jonker, and Y. Mohammad. [COMP24] the automated negotiating agents competition (ANAC) 2024 challenges and results. In S. Das, A. Nowé, and Y. Vorobeychik, editors, *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2025, Detroit, MI, USA, May 19-23, 2025*, pages 3000–3002. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2025. doi: 10.5555/3709347.3744072. URL https://dl.acm.org/doi/10.5555/3709347.3744072.

[4] T. Baarslag, R. Hadfi, K. V. Hindriks, T. Ito, and C. M. Jonker. Optimal non-adaptive concession strategies with incomplete information. In N. Fukuta, T. Ito, M. Zhang, K. Fujita, and V. Robu, editors, *Recent Advances in Agent-based Complex Automated Negotiation [revised and extended papers from the 7th International Workshop on Agent-based Complex Automated Negotiation, ACAN 2014, Paris, France, May 2014]*, volume 638 of *Studies in Computational Intelligence*, pages 39–54, Cham, 2014. Springer International Publishing. doi: 10.1007/978-3-319-30307-9_3.

[5] T. Baarslag, R. Aydogan, K. V. Hindriks, K. Fujita, T. Ito, and C. M. Jonker. The automated negotiating agents competition, 2010-2015. *AI Mag.*, 36(4):115–118, 2015. doi: 10.1609/AIMAG.V36I4.2609. URL https://doi.org/10.1609/aimag.v36i4.2609.

[6] L.-A. Busch and Q. Wen. Finite Horizon Negotiation Games. *Annals of Economics and Finance*, 2(2):415–435, November 2001. URL https://ideas.repec.org/a/cuf/journl/y2001v2i2p415-435.html.

[7] D. de Jonge. A new bargaining solution for finite offer spaces. *Applied Intelligence*, Sep 2023. ISSN 1573-7497. doi: 10.1007/s10489-023-05009-1.

[8] D. de Jonge. *Introduction to Automated Negotiation*. self-published, Barcelona, Spain, 2025. URL https://www.iiia.csic.es/~davedejonge/intro_to_nego.

[9] D. de Jonge. Bingo - supplementary material, 2025. URL https://www.iiia.csic.es/~davedejonge/.

[10] D. de Jonge, T. Baarslag, R. Aydoğan, C. Jonker, K. Fujita, and T. Ito. The challenge of negotiation in the game of diplomacy. In *Agreement Technologies, 6th International Conference, AT 2018, Bergen, Norway, December 6-7, 2018, Revised Selected Papers*, volume 11327 of *Lecture Notes in Computer Science*, pages 100–114, Cham, 2019. Springer International Publishing. ISBN 978-3-030-17294-7. doi: 10.1007/978-3-030-17294-7_8.

[11] K. Fujita, R. Aydogan, T. Baarslag, T. Ito, and C. M. Jonker. The fifth automated negotiating agents competition (ANAC 2014). In *Recent Advances in Agent-based Complex Automated Negotiation [revised and extended papers from the 7th International Workshop on Agent-based Complex Automated Negotiation, ACAN 2014, Paris, France, May 2014]*, volume 638 of *Studies in Computational Intelligence*, pages 211–224. Springer, 2014. doi: 10.1007/978-3-319-30307-9_13. URL https://doi.org/10.1007/978-3-319-30307-9_13.

[12] K. Fujita, R. Aydoğan, T. Baarslag, K. Hindriks, T. Ito, and C. Jonker. The sixth automated negotiating agents competition (anac 2015). In *Modern Approaches to Agent-based Complex Automated Negotiation*, pages 139–151. Springer, 2017.

[13] Y. K. Gal and L. Ilany. The fourth automated negotiation competition. In *Next Frontier in Agent-Based Complex Automated Negotiation*, volume 596 of *Studies in Computational Intelligence*, pages 129–136. Springer, 2015. doi: 10.1007/978-4-431-55525-4_8. URL https://doi.org/10.1007/978-4-431-55525-4_8.

[14] N. Gatti, F. D. Giunta, and S. Marino. Alternating-offers bargaining with one-sided uncertain deadlines: an efficient algorithm. *Artif. Intell.*, 172(8-9):1119–1157, 2008. doi: 10.1016/J.ARTINT.2007.11.007. URL https://doi.org/10.1016/j.artint.2007.11.007.

[15] F. D. Giunta and N. Gatti. Alternating-offers bargaining under one-sided uncertainty on deadlines. In G. Brewka, S. Coradeschi, A. Perini, and P. Traverso, editors, *ECAI 2006, 17th European Conference on Artificial Intelligence, August 29 - September 1, 2006, Riva del Garda, Italy, Including Prestigious Applications of Intelligent Systems (PAIS 2006), Proceedings*, volume 141 of *Frontiers in Artificial Intelligence and Applications*, pages 225–229. IOS Press, 2006. URL http://www.booksonline.iospress.nl/Content/View.aspx?piid=1680.

[16] F. D. Giunta and N. Gatti. Bargaining over multiple issues in finite horizon alternating-offers protocol. *Ann. Math. Artif. Intell.*, 47(3-4): 251–271, 2006. doi: 10.1007/S10472-006-9037-Y. URL https://doi.org/10.1007/s10472-006-9037-y.

[17] M. Graf, T. Engesser, and B. Nebel. Symbolic computation of sequential equilibria. In M. Dastani, J. S. Sichman, N. Alechina, and V. Dignum, editors, *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024, Auckland, New Zealand, May 6-10, 2024*, pages 715–723. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2024. doi: 10.5555/3635637.3662924. URL https://dl.acm.org/doi/10.5555/3635637.3662924.

[18] D. M. Kreps and R. Wilson. Sequential equilibria. *Econometrica: Journal of the Econometric Society*, pages 863–894, 1982.

[19] J. Mell, J. Gratch, T. Baarslag, R. Aydogan, and C. M. Jonker. Results of the first annual human-agent league of the automated negotiating agents competition. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents, IVA 2018, Sydney, NSW, Australia, November 05-08, 2018*, pages 23–28. ACM, 2018. doi: 10.1145/3267851.3267907. URL https://doi.org/10.1145/3267851.3267907.

[20] Y. Mohammad, E. Areyan Viqueira, N. A. Ayerza, A. Greenwald, S. Nakadai, and S. Morinaga. Supply chain management world - A benchmark environment for situated negotiations. In M. Baldoni, M. Dastani, B. Liao, Y. Sakurai, and R. Zalila-Wenkstern, editors, *PRIMA 2019: Principles and Practice of Multi-Agent Systems - 22nd International Conference, Turin, Italy, October 28-31, 2019, Proceedings*, volume 11873 of *Lecture Notes in Computer Science*, pages 153–169. Springer, 2019. doi: 10.1007/978-3-030-33792-6_10. URL https://doi.org/10.1007/978-3-030-33792-6_10.

[21] Y. Mohammad, S. Nakadai, and A. Greenwald. Negmas: A platform for automated negotiations. In T. Uchiya, Q. Bai, and I. Marsá-Maestre, editors, *PRIMA 2020: Principles and Practice of Multi-Agent Systems - 23rd International Conference, Nagoya, Japan, November 18-20, 2020, Proceedings*, volume 12568 of *Lecture Notes in Computer Science*, pages 343–351. Springer, 2020. doi: 10.1007/978-3-030-69322-0_23. URL https://doi.org/10.1007/978-3-030-69322-0_23.

[22] J. Nash. The bargaining problem. *Econometrica*, 18:155–162, 1950.

[23] M. J. Osborne and A. Rubinstein. *A course in game theory*. MIT press, 1994.

[24] B. Renting, D. de Jonge, H. Hoos, and C. Jonker. Analysis of learning agents in automated negotiation. Under review.

[25] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter*. The MIT Press, Cambridge, USA, 1994.

[26] A. Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109, 1982.

[27] R. Sloof. Finite horizon bargaining with outside options and threat points. *Theory and Decision*, 57(2):109–142, Sep 2004. ISSN 1573-7187. doi: 10.1007/s11238-004-4319-5. URL https://doi.org/10.1007/s11238-004-4319-5.