Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

Optimizing bid search in large outcome spaces for automated multi-issue negotiations using meta-heuristic methods

Mohammad Amini[a] and Mohammad Fathian[a*]

[a] *Department of Intelligent Systems, School of Industrial Engineering, Iran University of Science and Technology, Hengam St., Tehran, Iran*

C H R O N I C L E

Article history:
Received July 27, 2020
Received in revised format: October 23, 2020
Accepted October 25 2020
Available online October 25, 2020

Keywords:
Automated negotiations
Bidding Strategy
Outcome Space
Bid Search
Metaheuristics

A B S T R A C T

Bidding strategy is an important part of a negotiation strategy in automated multi-issue negotiations. In order to present good offers, which help maximize the agent's utility, we need to search the outcome space and find appropriate bids. Bid search can become challenging in large outcome spaces with more than ten thousands of possible bids. The traditional search methods such as exhaustive or binary search are not efficient enough to find the right bids in a large space. This is mostly due to the high number of issues, high number of possible values for each issue, and increased time complexity of usual search methods. In this paper, we investigate the potential of using meta-heuristic methods for optimizing bid search in large outcome spaces. We apply some of the most popular meta-heuristic algorithms for bid search in bidding strategy of baseline negotiating agents and evaluate their impacts on negotiation performance in different negotiation domains. The evaluation results obtained through comprehensive experiments show how meta-heuristic algorithms can help improve bid search capability and consequently negotiation performance of the agents on different performance criteria. In addition, we show which search algorithm is most suitable for improving any particular performance criterion.

1. Introduction

Automated negotiations is one of the main application areas of multi-agent systems in which intelligent and autonomous software agents negotiate on behalf of humans try to maximize their interests by taking into account their owner's preferences (Baarslag, 2016; Marsa-Maestre et al., 2014). In *bilateral multi-issue* negotiations, two intelligent agents, which may have contradictory interests, bargain over a set of issues to reach an agreement (Faratin et al., 1998). Each of the negotiation parties is willing to offer bids that bring the most amount of utility for them. In order to become successful, the negotiating agent must have an appropriate decision-making procedure for submitting offers. The basis of this procedure is the agent's *negotiation strategy* (Baarslag, 2016; Jennings et al., 2001). A very important component of a negotiation strategy is bidding strategy, (Baarslag, 2016; Baarslag et al., 2014). This component responsible to determine which offers the agent must send to its opponents and how to generate such offers so that the final gained utility is maximized. To solve this problem, the agent can apply a *negotiation tactic* or concession method. (Agrawal & Chari, 2009; Faratin et al., 1998). These concession methods use a decision function that maps each negotiation state into a target or expected utility value. When the target utility is determined, the agent performs a search process in the set of all possible offers to find a bid that has the closest utility to the target value. Therefore, the bid search problem is to minimize the distance between the utility of the found bid and the target utility value. Each negotiation has a certain deadline for completion and if the negotiations

* Corresponding author. Telephone: +9821-73225048, Fax: +9821-73021214
E-mail address: fathian@iust.ac.ir (M. Fathian)

© 2021 by the authors; licensee Growing Science, Canada.
doi: 10.5267/j.dsl.2020.10.007

are terminated without any agreement, the negotiating parties receive zero utility. Therefore, it is of high importance for an agent to find the optimal offer for each round of negotiation in a reasonable time.

Negotiations may take place in various domains that differ in the number of negotiation issues and possible options for each issue. As the value for these parameters increase, the domain size, which indicates the total number of possible offers will grow exponentially. The larger the outcome space, the longer it takes to find the right offer and this makes the agent's response slower in each round of negotiations. Taking more time to propose a bid will make the opponent wait more than expected and may lead to the response time-out error in the negotiation software framework. This challenge will be really serious if the negotiation is done in real-time. In addition, taking so long to find the right bid will make the negotiating agent send less bids during a certain period causing the reduction in the agent's bargaining power and loss of utility overtime. On the other hand, if the agent cannot find the desired offer in the large outcome space due to lack of time, then it may send non-optimal bids to its opponent and gain little utility in the negotiation session.

In general, the bid search in the outcome space is done using two common search methods: exhaustive search and binary search (Dowsland, 2014). In exhaustive search, every bid in the outcome space is examined and the optimal bid is returned as the best found offer. The time complexity of this search is equal to $O(n)$ in terms of the size of the outcome space if it includes n possible bids. For small domains, this is a sensible solution, because with a simple sequential search, the desired offer is obtained at an acceptable time. However, as the domain size increases, finding a solution with such a search will take a long time. Binary search is performed in less time, assuming that the set of possible bids is sorted based on the utility values. The optimal bid is found in $O(log\, n)$ time, which is less time compared to exhaustive search time. But to do this search, it is necessary to sort the outcome space at the beginning of the negotiation session. In most sorting algorithms the required time is as long as $O(n^2)$, although is some algorithms such as *Heapsort* and *Merge sort* the time complexity can be $O(nlog\, n)$ when using more memory space (Cormen et al., 2009; Knuth, 1998).

In most cases, automated negotiations are held in real-time or near real-time. Therefore, the negotiating agent has very little time to propose a bid in each round of negotiation. In the international Automated Negotiating Agents Competition (ANAC), which is held every year, the negotiation deadline is usually 180 seconds for both parties in the negotiation (Baarslag et al., 2015). In a domain that has as many as 10,000 possible bids, given there are 2,000 favorable bids for the agent to reach an agreement, the time available for the agent to find or generate each bid is about 0.045 seconds. This time is much less on domains with larger outcome spaces. Fig. 1 shows the outcome space for a negotiation domain with 59,049 bids. If the agent wants to find the desired bid with the target utility above 0.9, it must search for those bids in the blue areas. Considering the time required to implement the concession tactic and calculate the target utility in each round of negotiation, the time to search and find the desired offer in a large space will be much less than this. Thus, using a correct and efficient search method in large outcome spaces is critical for succeeding in the negotiations. The common search methods used for small domains such as exhaustive and binary search do not have enough efficiency in this area.

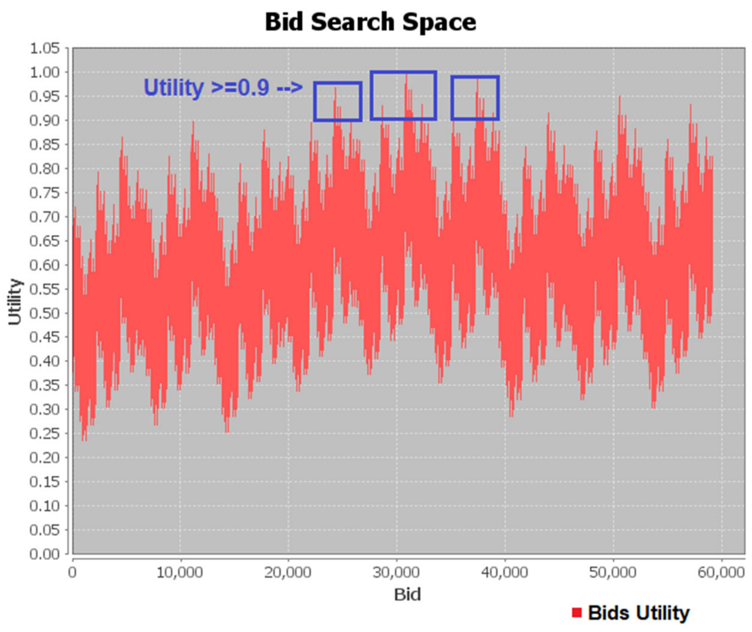

Fig. 1. The outcome space for a domain with 59,049 bids, the blue rectangles show the areas to search for the bids with $utility \geq 0{:}9$

A promising solution for efficient search in large space is to use meta-heuristic methods. Metaheuristics are a class of problem-solving methods that use a high-level strategy to guide and refine the search in an iterative manner to achieve an optimal solution (Burke, et al., 2014; Gendreau et al., 2019; Talbi, 2009). These methods allow us to solve large-scale optimization problems by finding satisfactory solutions in a reasonable time. Finding the desired bid for a negotiating agent through a large outcome space with respect to the target utility is an optimization problem that can be solved effectively using meta-heuristic algorithms. The required search time to solve a given problem is an important issue in the selection of an optimization algorithm. One of the important applications of meta-heuristic methods is when an instance of a polynomial problem has a large size so that it is not possible to solve it in real time by using exact methods or exhaustive search (Talbi, 2009). Indeed, even if the problem is polynomial, the need of using meta-heuristic may be justified for real-time search constraints. According to what mentioned about the performance of exhaustive and binary search, using meta-heuristic methods to search for a bid in a large outcome space will be useful.

In this paper, we intend to investigate and analyze the potential of using meta-heuristic methods in automated negotiations with large outcome space and to examine their effect on bid search in a systematic way. The contribution of this article is presented in two parts: First, the well-known meta-heuristic methods from different families have been implemented as a search module for finding appropriate bids in the negotiating strategy of some baseline negotiating agents. We describe the implementation method of metaheuristics for bid search and how to manage different parameters of the algorithms. Second, by conducting purposeful and systematic experiments and examining various performance measures in negotiation, we perform a comparative study on all implemented meta-heuristics and determine the effectiveness of each method on improving the bidding strategy of negotiating agents. Through this comparison, we can introduce the best and most appropriate meta-heuristic methods to improve each performance criteria in negotiation.

The rest of this paper is organized as follows: In section 2, we present a background on automated negotiation concepts and related work in the use of metaheuristics. Section 3 includes our research methodology for implementing meta-heuristic algorithms for bid search in the negotiating agents, the description of comparative study, and experiment setups. In section 4, we present the results and analysis. Section 5 concludes the paper and suggests some appropriate future directions.

2. Background and Related Works

In this section we provide a background about the bilateral automated negotiations and a review on the applications of metaheuristics in the automated negotiations.

2.1. *Automated Negotiations*

In a bilateral automated negotiation, two autonomous agents negotiate about one or more *issues* until they reach an agreement which is satisfactory for both agents (Baarslag, 2016; Faratin et al., 1998). The negotiation domain specifies the issues that are going to be negotiated and the available options or values for each issue. If the negotiation domain (*D*) includes a set of *n* issues such as $(I_1, I_2, \dots, I_n)$, each issue I_i can have a set of values or options as $\{a_1, a_2, \dots, a_{m_i}\}$ and $m_i \in N$ is the number of values or options for the issue I_i. A mapping of each issue to one of its values or options is called a bid or outcome (ω). The set of all possible bids in the domain is called the outcome space (Ω). The size of the outcome space also known as domain size is affected by the number of issues and their options. Therefore, the size of the negotiation domain (*D*) can be calculated as $\prod_{i=1}^{n} m_i$.

The two agents negotiate with each other by proposing the bids that are more interesting for them. An *agreement* on *n* issues $(I_1, I_2, \dots, I_n)$ is an outcome in the form of $\omega = (v_1, v_2, \dots, v_n)$ which is accepted by both agents, where v_i is the agreed value for the issue I_i. Each agent might have different preferences over the issues as well as options for each issue. The set of all preferences of the agent is called the agent's preference profile. This preference profile can be full or partial depending on the preference relations that the agent considers on the domain. The most common way of creating a preference profile is by assigning weights to the issues and defining utility for each issue value. Therefore, a utility function is defined to map each bid in the outcome space to a real value that shows the amount of desirability of the bid for the agent. A utility function can be defined in linear or non-linear forms (Baarslag et al., 2015; Jennings et al., 2001), but in most studies the linear additive utility has been used. In this type of utility, the total utility of a bid is the sum of the weighted utilities of all the assigned issue values in the bid.

The deadline of a negotiation denotes the point of time before which an agreement must be reached. The deadline may be specified as the maximum number of negotiation rounds or alternatively as a real-time target. The interaction between agents in a negotiation session is specified by a *negotiation protocol*. It defines a set of rules that determine how and when agents will make their own offers and what options they can take when negotiating. The most common negotiation protocol used in bilateral negotiations is called alternating offers protocol (Osborne et al., 2004). According to this protocol, the two negotiating agents propose their bids in turns. If the other party likes the offer, it will send an accept message. If it does not accept the offer, it will send its own offer as a counter-offer to the other party. This process is repeated until the agents reach an agreement or the negotiation deadline arrives. If the deadline is reached and no agreement has been found, the negotiation is failed and the agents are said to face a break-off.

The important challenge for a negotiating agent is to find out which bid to propose in what point in time in order to gain the most amount of utility out of negotiation. To deal with this challenge, a very broad research direction has been established in the community to work on negotiation strategy (Baarslag, 2016; Baarslag et al., 2015; Faratin et al., 1998). A negotiation strategy forms the foundation for an agent's decision-making during negotiation. Negotiation strategy can be decoupled into three important components each of which contributes to the agent's overall decision making process (Amini et al., 2020; Baarslag, 2016; Baarslag et al., 2014): bidding strategy, opponent model, acceptance strategy. Bidding strategy is the main component which determines a concession behavior during negotiation and specifies how to generate appropriate bids according to certain negotiation parameters. Generally, there are two types of concession behavior or negotiation tactics: *time-based* tactics and *behavior-based* tactics (Baarslag, 2016; Faratin et al., 1998). Time-based tactics use a decision function that returns a target utility for each point in time. The time-based decision function determines the speed of concession during negotiation. For example, *Boulware* Strategy uses a time-based tactic which concedes strictly at the beginning of negotiation while *Conceder* Strategy concedes more in the initial rounds to avoid breakoff (Baarslag, 2016). Behavior-based tactics define the target utility according to the opponent's concession behavior (Baarslag et al., 2014; Faratin et al., 1998).

2.2. Meta-heuristics for automated negotiations

Metaheuristics are generally a family of approximate optimization techniques that orchestrate an interaction between local improvement procedures (heuristics) and higher level strategies in order to find global optimal solutions for a problem (Gendreau et al., 2019; Talbi, 2009). Although finding global optimal solutions is not always guaranteed, they can provide "acceptable" solutions in a reasonable time for solving complex problems by efficiently exploring the search space (Talbi, 2009).

Meta-heuristic methods can be classified based on the multiplicity of solutions that are manipulated in the search guiding process. In this regard, meta-heuristics algorithms are categorized as single-solution based and population-based methods (Gendreau et al., 2019; Talbi, 2009). In Single-solution based algorithms such as Local Search (Hoos & Stützle, 2004), Simulated Annealing (Kirkpatrick et al.,1983) and Tabu Search (Glover, 1997), a single solution is manipulated or transformed during the search. The population-based algorithms guide the search procedure by working on a number of solutions (called population) sampled from the search space and evolve them through the search until they include acceptable solutions. Evolutionary algorithms such as genetics algorithms (Mitchell, 1998) and swarm intelligence techniques including ant colony optimization (Dorigo & Blum, 2005) and particle swarm optimization (Kennedy & Eberhart, 1995) are considered as population-based methods. In recent years, many nature-inspired metaheuristic methods have been developed (Yang, 2017), including Bat Algorithms, Bee Colonies and Artificial Immune Systems.

So far, a few cases of using meta-heuristics for bidding strategy have been observed in the automated negotiation research community. Especially, in ANAC 2014 (Fujita et al., 2016), which was held with the subject of negotiation over large spaces, a number of successful participant agents used meta-heuristic algorithms to explore the outcome space in order to find the desired offers. *AgentM* (Niimi & Ito, 2016), which won first place, used simulated annealing (SA) (Kirkpatrick et al., 1983) to find the best offer according to the target utility and then it modifies this offer with regard to the best offer received from the opponent so that it is also desirable for the opponent. The agent *GANGSTER* (De Jonge & Sierra, 2016), which ranked second, used genetic algorithm (GA) (Mitchell, 1998) in its bidding strategy to search for the best possible bids. The *WhaleAgent* (Sato & Ito, 2016) uses two methods to search the outcome space: SA (Kirkpatrick et al., 1983) and hill climbing (local search) (Hoos & Stützle, 2004). The agent first starts the hill climbing using the best offer received from the opponent and tries to perform a local search to find the local optima near this offer. If a bid with a utility value greater than the target utility is not found, then the agent uses the SA search method with a random start to find a global optimum. The use of local search is also seen in *AgentYK* (Kadono, 2016. Recently, a combination of owl search algorithm (OSA) (Jain et al., 2018) and chaos theory has been used to find the optimal offers for the negotiating agent (El-Ashmawi et al., 2020). In this method, a population of bids evolves toward an appropriate bid with enough utility.

Although efforts have been made to use meta-heuristic algorithms to search for offers in the outcome space, there are two major gaps in this area. First, the applications of meta-heuristic methods in this area are not numerous and widespread enough to accurately show the effect of meta-heuristics from different families on bid search and bidding strategy improvement. Second, any research study, which has applied a particular meta-heuristic method (e.g. GA or SA) in automated negotiation bidding strategy, has considered it as one possible method to improve search and did not mention the scientific or experimental reason for using such method. In addition, most of the time in the evaluations, the applied method has not been comprehensively compared with other similar meta-heuristics to demonstrate its real performance. Therefore, the question remains as to what meta-heuristic approach is appropriate for bidding in bilateral negotiations and what are the criteria for choosing a search method? In this article, we intend to provide answers to these questions with a systematic examination of the effect of meta-heuristic methods in bid search and evaluating these methods based on various performance criteria in negotiation.

3. Research Methodology

In this section, we present our research methodology used to examine the potential of using metaheuristic algorithms for bid search in automated negotiations. Overall, our study has three specific objectives:

- Objective 1: Investigating the meta-heuristics' effect on bid search (accuracy of finding the optimal bid according to the target value)
- Objective 2: Determining the effect of different meta-heuristic algorithms on negotiation performance
- Objective 3: Identifying the best meta-heuristic algorithms for improving certain negotiation performance criteria

To achieve these objectives, a systematic and comprehensive methodology based on numerous experiments on different domains has been implemented. Our method for this study is as follows: Two baseline simple bidding strategies are selected as the implementation platform for search methods. Next, we implement some well-known meta-heuristic algorithms from different families in the search module of these base strategies. Then, by using various experiments in the form of negotiations in the context of different domains, we compare the effect of these methods on the quality of selected desired offer and the success rate of that strategy in negotiations. Various performance metrics are used to perform comparisons.

In the continuation of this section, first, we describe the bilateral multi-issue negotiation model and the agent's decision framework. Then, the baseline bidding strategies are introduced and the method of implementing meta-heuristic algorithms in bid search for those baseline strategies are explained. Next, we introduce the meta-heuristic algorithms selected for implementation as search methods. Finally, we describe the structure of experiments that are conducted to measure the effect of meta-heuristics on negotiation performance.

3.1. Negotiation Model

Fig. 2 shows the decision structure the agent in dealing with its opponent in a bilateral negotiation session. The two negotiating agents operate on the basis of *alternating offer protocol* (Osborne et al., 2004). The negotiation domain includes *n* issues in the set $\boldsymbol{I} = (\boldsymbol{I_1}, \boldsymbol{I_2}, \dots, \boldsymbol{I_n})$ and each issue $\boldsymbol{I_i}$ can be assigned $\boldsymbol{m_i}$ number of values or options as displayed by the set $\boldsymbol{A_i} = (\boldsymbol{a_1^i}, \boldsymbol{a_2^i}, \dots, \boldsymbol{a_{m_i}^i})$. Each agent makes an offer or $\boldsymbol{bid(\omega)}$ to its opponent in each round of negotiation:

$$\omega = (\phi_1, \phi_2, \dots, \phi_n) \quad , \phi_i \in A_i. \tag{1}$$

If the offer is acceptable to the opponent, it agrees with it by sending an "Accept" message. Otherwise, the opponent sends a *counter-offer* which is also made in the form of eq. 1.

From our negotiating agent's point of view, the following task are done in each round of a negotiation session:

a) The *concession behavior* component determines the *target utility*, based on a *decision function*. As mentioned in the previous section, the decision function can be time-based or behavior-based. In this paper, we consider a standard time-based function $\Psi: [\mathbf{0}, \mathbf{1}] \to [\mathbf{0}, \mathbf{1}]$ as follows (Fatima et al., 2002):

$$\psi(t) = Q_{\min} + (Q_{\max} - Q_{\min})(1 - \Gamma(t)), \tag{2}$$

in which Constants Q_{min} and Q_{max} $\in [0,1]$ control the allowed range of utility values for the bids and the time-dependent function $\Gamma(t)$ is defined as follows:

$$\Gamma(t) = \vartheta + (1 - \vartheta) t^{\frac{1}{\alpha}}, \quad t \in [0,1]. \tag{3}$$

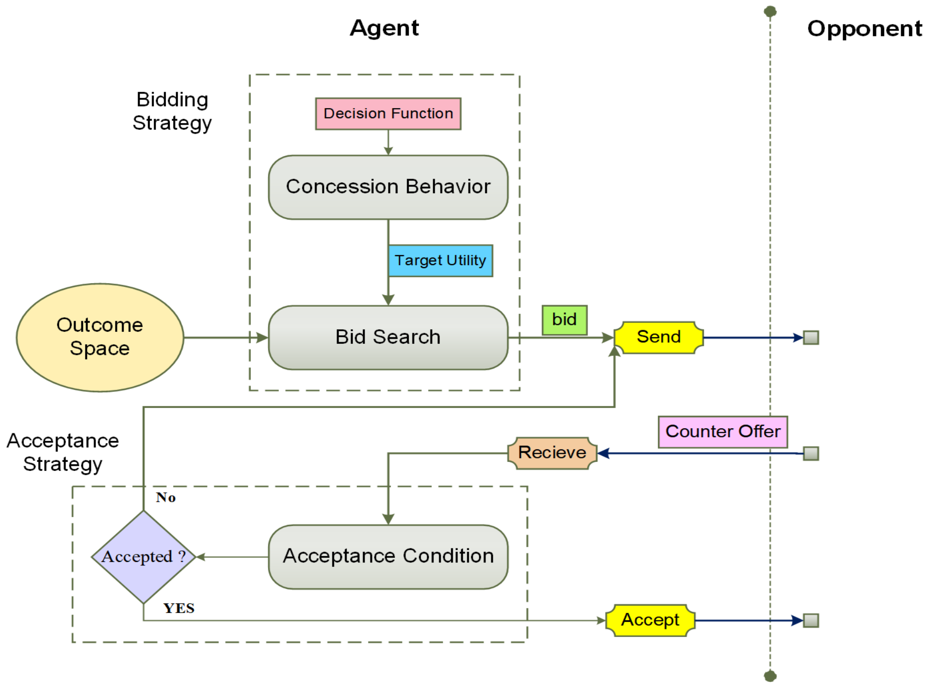

Fig. 2. The architecture of a negotiating agent with bidding and acceptance strategies

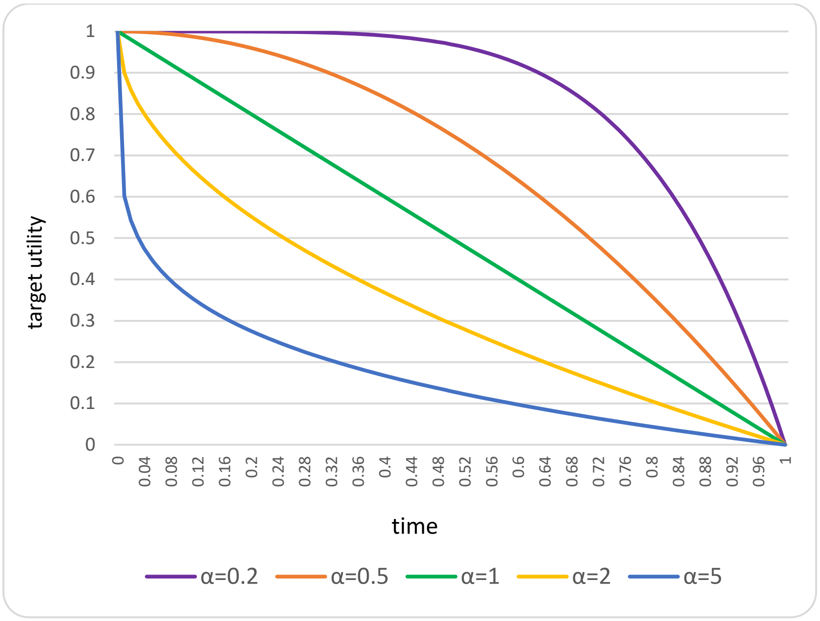

Fig. 3. Target utility values through the negotiation time using different time-based tactics (Baarslag, 2016)

The negotiation time is normalized in the interval$[\mathbf{0},\mathbf{1}]$. In eq. 3, the value must be small enough in a way that the function $\Gamma(\mathrm{t})$ outputs a small value in $t = 0$ and the target utility at the beginning of the negotiation be close to Q_{max} . The parameter α determines the concession style of the agent and can be changed to create different concession tactics accordingly. Fig. 3 depicts the changes in target utility during the negotiation time in terms of different values for α. If $0 < \alpha < 1$ the agent uses a strict strategy called *Boulware* (Baarslag, 2016; Faratin et al., 1998). This strategy concedes hardly during the negotiation and only in the final rounds of the negotiation it increases the concession rate. When $\alpha > 1$, the agent adopts a *conceder* strategy that compromises more from the beginning of the negotiation (Baarslag, 2016; Faratin et al., 1998). This strategy is useful to prevent break-off in the negotiation. In the case of $\alpha = 1$, the agent concedes linearly according to time. This strategy is called *Linear Conceder*.

b) The search component of the bidding strategy tries to find the bid with a utility that is closest to the target value. The calculation of the utility for a bid is based on a utility function that maps each bid (ω) in the outcome space to a real value in the interval [0,1]. This study uses the additive utility function (Baarslag, 2016; Raiffa, 2007), in the form of

$$U(\omega) = \sum_{i=1}^{n} w_i e_i(\phi_i), \quad \phi_i \in A_i, \quad i \in \{1, 2, ..., n\}, \quad \sum_{i=1}^{n} w_i = 1, \tag{4}$$

where w_i is the normalized weight of the issue I_i , indicating the importance of this issue for the agent, and the e_i is an evaluation function for showing the agent's evaluation of each option assigned to the *i-th* issue.

c) The agent negotiation strategy also uses an acceptance condition that determines how the agent accepts a particular bid. One of the most common acceptance criteria is comparing the utility of the received offer with the target utility or the utility of the offer that the agent is ready to send in the current time (Baarslag et al., 2014; Fatima et al., 2002). The acceptance condition in the time t, for the current received bid from the opponent (ω^c) is defined generally as:

$$AC(\omega^C, t) \Leftrightarrow \mu . U(\omega^C) + \xi \geq \Psi(t), \tag{5}$$

in which μ is a scale factor and ξ can be considered as a utility threshold that defines the acceptable gap between the opponent's bid utility and the target utility value. By setting $\mu = 1$ and $\xi = 0$ we implement a simple acceptance condition for our agent which allows the acceptance if the utility as follows:

$$AC(\omega^C, t) \Leftrightarrow U(\omega^C) \geq \Psi(t). \tag{6}$$

A complete review of the acceptance conditions in the negotiation strategy can be found in (Baarslag et al., 2014). If the opponent's offer is accepted according to the acceptance condition, the "Accept" message will be sent to it. Otherwise, the bid which is produced by the bidding strategy will be sent to the opponent.

Algorithm 1 shows the decision-making procedure of our agent in a round of negotiation. The input of this algorithm is the opponent's bid (ω^c), the current time (t) and outcome space (Ω) while its output is a new found bid (ω^n) or an "Accept" message. u^T denotes the target utility. This algorithm implements a baseline operation mode for the negotiating agent. To evaluate the performance of meta-heuristic methods, we apply meta-heuristic algorithms to the *search function* of the agent shown in line 2 of the algorithm.

In the bid search component, different methods can be used to perform the search. Common methods for search in the outcome space are exhaustive search and binary search (Burke et al., 2014; Dowsland, 2014). Meta-heuristics can be good candidates for improving the efficiency of the bid search, which is the subject of this article.

Algorithm 1: Agent's functional procedure in a round of negotiation

```
Input: opponent's bid ( $\omega^c$ ), $t, \Omega$
Output: A new Bid ( $\omega^n$ ) or 'Accept'

1: $u^T$= biddingStrategy.targetUtility(t)      //according to Eq. (2)
2: $\omega^n$ = biddingStrategy.Search($u^T, \Omega$)
3: $u_1$ = calculateUtility ( $\omega^c$ )     //according to Eq. (4)
4: $u_2$ = calculateUtility ( $\omega^n$ )     //according to Eq. (4)
5: If ( $u_1 \geq u_2$ )
6:     send('Accept')
7: else
8:     send( $\omega^n$ );
```

3.2. *Baseline bidding strategies*

In order to evaluate and benchmark the performance of meta-heuristic algorithms in the bid search component, two basic agents with simple baseline strategies are used: a) The *Boulware* strategy which has a stubborn concession behavior that concedes only at the end of the negotiation, b) *Linear Conceder* which uses a linear concession rate throughout the negotiation. The negotiation method of these two agents are the same and according to Algorithm 1, with the difference that their concession behavior is different. We described these two strategies in the previous subsection. Both strategies use a time-dependent decision function according to Eq. 2 and Eq. 3. But in the Eq. 3, we set $\alpha = 0.2$ for the *Boulware*, and $\alpha = 1$ for the *Linear Conceder* (see Fig. 3). The reason why we did not chose a Conceder strategy with $1 < \alpha$ is that this strategy concedes too early in the negotiation (e.g. before $1/10$ of the negotiation time passes) and reaches to agreement with little utility. Therefore, it cannot give us a good evaluation about the effect of the meta-heuristics on the negotiation performance.

3.3. Meta-heuristics Implementation

In order to implement meta-heuristic algorithms for the bid search in the outcome space, we must define the problem to be manageable and solvable for these algorithms. Despite the differences in the solution approach, meta-heuristic algorithms also include common design questions to be answered. We need to answer these design questions to prepare the bid search problem for the implementation of meta-heuristic algorithms. To be more specific, we need to describe *solution representation*, *neighborhood structure*, and *objective function*.

Solution representation: The negotiation outcome space is the search space, and each bid represents a candidate solution. Therefore, we must encode each bid appropriately for the use of meta-heuristic algorithms. In a negotiation domain with *n* issues, each bid (as in Eq. 1) can simply be displayed as a vector b with n elements of discrete values as follows:

$$\omega = (\phi_1, \phi_2, ..., \phi_n) \rightarrow b = < b_1, b_2, ..., b_n > \quad , \phi_i, b_i \in A_i \quad , i = 1, 2, ..., n. \tag{7}$$

With this representation the outcome space (Ω) is also mapped to the set of *n*-element bid vectors bids (B). It is worth noting that the values or options for each issue are nominal (eg. days of the week, country names, or hours of the day). Therefore, no mathematical calculations can be done on these values. However, with the use of the evaluation function as in Eq. 4, we can have a quantified real-number evaluation of an issue value for the agent.

Neighborhood Structure: One of the important things in designing a search procedure with metaheuristics is determining how to navigate the search space to find other solutions. Movement in the search space is done using a well-defined neighborhood structure. The neighborhood of a solution is obtained based upon a neighborhood function (Gendreau et al., 2019; Talbi, 2009). In a search space S, a neighborhood function is a mapping like $N: S \rightarrow 2^S$ which maps each solution $s \in S$ to a set of solutions $N(s) \subset S$ that are reachable from s. The neighborhood structure in the problem of searching the outcome space is defined for each bid ($b \in B$) as follows:

$$N(b) = \{c \in B \mid c =< c_1, c_2, ..., c_n >, \forall i\{1, 2, ..., j-1, j+1, ..., n\} : \\ c_i = b_i,\ c_j = a \in A_i, a \neq b_j\} \quad (8)$$

In fact, each neighbor of a bid in the search space can be generated by replacing the value of *only one* issue in the current bid by one of the possible values in the issue values set. The neighborhood size, or the number of possible neighbors for a bid, in a domain with n issues can be obtained by:

$$|N(b)| = \sum_{i=1}^{n} m_i - 1, \quad (9)$$

where m_i is the number of possible options for each issue I_i. For example, for a domain with 10 issues and 3 options for each issue, the size of the neighborhood for each bid is 10×2 = 20. Depending on the type of meta-heuristic algorithm, we can use a subset or all of the neighbors in $N(b)$ for exploration in the search space.
In order to implement meta-heuristic algorithms, we need to automatically generate the neighborhood structure for each bid in the search iterations. Algorithm 2, shows the procedure for generating the neighborhood of a bid during one iteration of a meta-heuristic algorithm.

Objective function: An objective function actually represents the quality or fitness of any possible solution. For our problem, it can be defined as the function $f: B \to \mathbb{R}$ which maps any bid vector from the outcome space to a real value and thus shows the value of the solution in the search process. In the negotiation, the utility function mentioned in Eq. 4 is a function that shows the quality of each bid in the agents' preference profile. However, this utility function provides an absolute value in the range [0,1] in terms of agent utility space for each bid. Whereas, our task is to find a bid vector that is closer to a declared value of utility (target utility). For this purpose, the following objective function is defined:

$$f(b) = \left| U(b) - u^T \right|, \quad (10)$$

in which $U(b)$ is the utility value for the bid *b* in the agent's utility space, calculated by Eq. 4, and u^T is the target utility.

Algorithm 2: Generating the neighborhood for a bid

```
Input: a bid ($b = < b_1, b_2, ..., b_n >$), The set of issue values $A = \{A_i\}$
Output: list of neighbors of the bid ($N(b)$)
1: Set $N(b) = \emptyset$
2: For each $b_i$ in $b$ Do
3: set $a_k = b_i$   // The value assigned to issue $I_i$
4:     For each $a_j$ in $A_i$ , ($a_j \neq a_k$) Do
5:         set $b' = b$      // creating a new bid
6:         set $b'_i = a_j$  // $b'_i$ is the value of $I_i$ in $b'$
7:         add $b'$ to $N(b)$
```

3.4. Meta-heuristic algorithms

To investigate the effect of meta-heuristic algorithms on the search efficiency and bidding improvement in the agent's bidding strategy, we have used some of the most important algorithms applied in research community to solve important optimization problems. We have made attempts to vary the algorithms in terms of the type of manipulation operation, the used heuristics and the method of achieving the final solution. Accordingly, seven meta-heuristic algorithms were chosen to be used:

- Two algorithms based on simple local search: the common local search (LS) or hill climbing (Hoos & Stützle, 2004) and variable neighborhood search (VNS) (Mladenovic & Hansen, 1997)
- Two simulated annealing-based algorithms: stochastic Simulated Annealing (SA) (Kirkpatrick et al., 1983) and Threshold Accepting (TA) (Dueck & Scheuer, 1990) which is the deterministic type of SA
- Taboo search (TS) algorithm which is a meta-heuristic algorithm with memory (Glover, 1997)

- A Genetic algorithm (GA)(Davis ,1991) which is an evolutionary type of metaheuristic
- Ant colony optimization (ACO) (Dorigo & Blum, 2005) algorithm which is a nature-inspired meta heuristic based on swarm intelligence.

Among the above seven algorithms, five are single solution-based and two are population-based algorithms. In the following, we briefly introduce each of the selected meta-heuristic algorithms:
Local search (Hill Climbing): It is the simplest type of search algorithm which starts with one random solution and in each iteration, a neighbor with better quality in terms of objective function replaces the current solution (Hoos & Stützle, 2004). n order to implement local search in this article, we use the best improvement method. That is, in each iteration, we choose the solution that has the *best improvement* in the objective function value in the neighborhood compared to the others. This method requires a complete exploration of the neighborhood but can reduce the number of iterations in the entire search.

Variable Neighborhood Search: It is a stochastic algorithm that starts with a random solution and explores different predefined neighborhood structures with local search to find the global optima (Mladenovic & Hansen, 1997). In fact, this algorithm uses variable neighborhoods as a mechanism to escape from the local optima. Neighborhood change for a solution occurs if local search does not improve the objective function.

Simulated Annealing: It is a stochastic algorithm inspired by the annealing process in mechanics. In this algorithm, the solutions which do not improve the objective function are accepted with a special probability that is a function of the temperature and the amount of degradation by those solutions (Kirkpatrick et al., 1983). This allow us to skip the local optima. The algorithm starts from a random solution, in each iteration a random neighbor is generated. If the neighbor improves the objective value compared to the current solution, it is always replaced with the neighbor. If the neighbor is not better, it will be accepted with a probability that is dependent on the difference between the objective value between the current solution and the generated neighbor. As the algorithm progresses, the probability of accepting bad moves decreases.

Threshold Accepting: This method is the deterministic version of the SA algorithm (Dueck & Scheuer, 1990). In this algorithm, the neighbors worse than the current solution are accepted if they do not degrade the value of the objective function more than a specified threshold. TA is a faster algorithm than SA because calculating random numbers and exponential functions takes up a considerable amount of computational time. So far, some studies have shown that this method is better at solving combinatorial optimization problems such as TSP compared to SA (Dueck & Scheuer, 1990; Mitra et al., 1985). Therefore, we used this algorithm together with the SA to investigate the effects of being deterministic and fast on the bid search process.

Tabu Search: It is a meta-heuristic method which uses memory to store search-related information (Glover, 1997). Taboo search method works like a local search algorithm with the steepest descent, but when it gets stuck in the local optima, it also accepts non-improving solutions in the neighborhood. This may result in a loop. Therefore, a list is used to store the solutions encountered so far, which is called the *taboo* list. The solutions in the list are not selected or used unless they meet an aspiration criteria. The taboo list is a short-term memory, so it has a limited size and is updated in each search iteration. This search method can also use medium-term memory to store elite solutions (best visited solutions) as well as long-term memory (all visited solutions) (Talbi, 2009). The medium-term memory is applied for the *intensification* process, that is, to limit the search to the features of elite solutions. The long-term memory is used for *diversification* which means to extend the search into undiscovered spaces.

Genetic algorithms: These stochastic algorithms are a well-known class of evolutionary algorithms. A genetic algorithm starts with a set of randomly selected solutions as the initial population. In each iteration, the current population is manipulated using crossover and mutation operators (Davis, 1991). In order to do the crossover, a portion of the population is selected as parents using a selection schema. Solutions with better fitness are more likely to be selected for reproduction. In GA, the crossover operator operates on two individuals and the content of the genes or constituent elements of the solutions are switched based on a specific pattern. The mutation operator also randomly alters the content of an element in a population. After the reproduction takes place with the operators, a replacement strategy is used to determine which members of the existing population should remain in the new population. The process of selection, reproduction and replacement is performed in different iterations to meet the stopping criteria.

Ant colony optimization: This method is one of the most successful meta-heuristics inspired by swarm intelligence (Kennedy, 2006). The algorithm takes the idea from the ants cooperative behavior pattern to perform complex tasks (Dorigo & Blum, 2005). Each ant creates a solution in a greedy stochastic process with the incremental addition of elements. The creation of each solution updates a shared data structure named the pheromone matrix. This matrix represents the characteristics of good solutions. The pheromones guides the process of making solutions by ants. For the negotiation problem, we consider the $n \times m$ pheromone matrix (λ) to maintain the information on the quality of the bids that have been created by ants. In fact, if the distance of a bid's utility to the target utility is smaller, then the amount of pheromone value is more. We also use the *frequency of the occurrence* of an issue value in the best created bids as a heuristic to create the new bids. The reason to do this is the higher the frequency of an issue value is in the best found bids, the better solutions will be generated using these values. The pheromone matrix is updated in two ways: evaporation and reinforcement. In the evaporation phase, the pheromone is reduced equally by a fixed proportion. In the reinforcement phase, the pheromone is increased for the elements of the solutions that are created.

3.5. Evaluation Method

Our evaluation method is based on multiple experiments in the form of automated negotiations in the context of different domains to evaluate specific aspects of performance in negotiation. To perform the experiments, we implement each of the meta-heuristic methods described in section 3.4, as well as two common search methods (exhaustive search and binary search) in the bid search component of the bidding strategy of an agent that uses one of the two baseline strategies described in section 3.2. Table 1 shows the 9 search methods that we used in our experimental evaluations. We examine the effect of different search methods on the accuracy of the search and the effectiveness of negotiation strategy in real negotiation environment. By placing the two traditional search methods within the experiments and comparing their performance with meta-heuristic methods, we can better understand the effect of meta-heuristics on the search and their improvements in the negotiation performance in a large space.

The Evaluation of different search methods are performed separately for *Boulware* and *Linear Conceder* bidding strategies. Because these two strategies have two different time-dependent concession tactics. To evaluate the meta-heuristics without bias, the performance of each search method that is combined with a concession tactic should be compared with another one that has been implemented alongside a similar concession tactic. Accordingly, two categories of experiments are always performed at each stage of the evaluations:

a) Implement search methods on Boulware strategy and evaluate them
b) Implement search methods on Linear Conceder strategy and evaluate them

The experiments results are presented separately for each of the categories above. For each category of experiments, there are 9 agents which are made with the baseline strategy and one of the 9 search methods that were introduced previously (2 basic search methods and 7 meta-heuristics).

Table 1
Different search methods that are evaluated in the experiments

Type of search	Algorithm	Abbreviation
Basic search	Exhaustive Search	ES
	Binary Search	BS
Metaheuristic	Local Search (Hill Climbing)	LS
	Variable Neighborhood Search	VNS
	Simulated Annealing	SA
	Threshold Acceptance	TA
	Tabu Search	TS
	Genetic Algorithm	GA
	Ant Colony Optimization	ACO

Table2
Three negotiation domains used in the experiments

Domain name	Number of Issues	Number of values for each issue	Outcome Space size (# bids)
D1	10	3	59,049
D2	8	5	390,625
D3	10	4	1,048,576

In order to make an accurate and non-biased assessment of the search methods, the experimental negotiations also take place in three large negotiation domains of different sizes. Table 2 shows the three negotiation domains with different outcome space sizes. It should also be noted that in each domain, two different preference profiles have been defined that shows the preferences adopted by the two agents negotiating with each other. For each negotiation domain, the two agents negotiate twice with each other so that each agent plays both preference profiles. In order to make a robust and reliable evaluation and to achieve research objectives in evaluating metaheuristic methods, three stages of experiments are performed:

Experiment A: Investigating the effect of meta-heuristic methods on bidding strategy

In this stage, 6 agents with baseline strategies are used as benchmarking opponents. The agents created by each of the 9 search methods mentioned in table 1 negotiate with each of these 6 opponents on the three existing negotiation domains. Table 3 shows the six benchmarking opponents for this stage of experiments. These benchmark agents are made as follows:

Baseline strategy + one basic search method (*binary search and exhaustive search*)

Our baseline strategies are of three types: Boulware, Conceder, and Linear Conceder. In this way, six benchmark bidding strategies are obtained that can be the basis for comparing the performance of the 9 agents with different search methods. Each agent with one of the 9 search methods negotiates against these six opponents. As a result, 54 negotiation encounters are performed for each category. Considering two negotiation sessions between the agents for each negotiation domain (by swapping the preference profiles), the total number of negotiation sessions for the three domains for each category is equal to 3×2×54 = 342.

Table 3
Benchmarking opponents for experiment A

Opponent number	Search Method	Concession Method
1	Exhaustive Search	Boulware
2	Binary Search	Boulware
3	Exhaustive Search	Conceder
4	Binary Search	Conceder
5	Exhaustive Search	Linear Conceder
6	Binary Search	Linear Conceder

Our goal at this stage of the experiments is to test the performance of meta-heuristic methods compared to the usual search methods used in the literature, i.e. exhaustive search method and binary search, and to examine the effect of meta-heuristics on search improvement. Their average performance of each agent in six negotiation sessions is obtained based on negotiation performance criteria and compared with that of the others. Negotiation performance criteria are introduced section 3.6.1 and Appendix A.

Experiment B: Studying the accuracy of search methods in finding the best offer

To avoid confusion and reduce computational volume, in this step we select six search methods from the top methods in stage A as the *selected* methods. In this step, we intend to evaluate the accuracy of a search method in finding a bid with the desired utility. To this end, we combine each of the selected search methods with the *Boulware* strategy and place it against a negotiating opponent which has a strict strategy and never accepts the offer. The reason for considering such an agent as the opponent is that if the offers are not accepted by it, the negotiation will last until the negotiation deadline and the evaluated negotiator agent will search and offer bids until the end of the negotiation. Therefore, a better evaluation can be obtained about the accuracy of the search in this agent. In addition, for the agents being evaluated, we only use the *Boulware* strategy because the Conceder methods end the negotiation session very quickly with compromise. Our goal in this experiment is to examine the gap between the target utility and the utility of the bid which is found by a search method. For this purpose, the 6 agents in the *Boulware* category negotiate with the non-accepting agent only once. Therefore, for each domain, 6 negotiation sessions are held. The total number of sessions in the three domains are 18.

Experiment C: Investigating the effect of meta-heuristics on overall negotiator's performance

In this stage, we will put the 6 search methods selected in Experiment B under a thorough evaluation using multiple negotiations in the form of a *Negotiation League*. To do this, each of the 6 methods is combined with Boulware and Linear Conceder strategies. As in Experiment A, negotiations take place in two separate categories. For each category of agents, a negotiation league is held on each of the three negotiation domains. During each league, each agent negotiates with other peers in two sessions by changing the preference profile. The results of the negotiations over each domain are presented according to the negotiation evaluation criteria (See Appendix A).

Our goal in this stage is to examine the overall effect of meta-heuristics on the performance of a negotiating agent in the negotiation environment and how meta-heuristic methods can improve the negotiation quality of an agent in different aspects according to performance criteria. In this way, good meta-heuristic methods for improving each negotiation performance criterion are identified. Moreover, because the agents face each other directly in a league, we can suggest the best negotiating agent in the league as a novel negotiation strategy to negotiate in a large space using an efficient search algorithm. To conduct a league in each category, each agent holds 5× 2 negotiation sessions with the other category members. The number of negotiations for each league on a domain is equal to 30 sessions and the total number of negotiation sessions held for each category is 30×3 = 90.

3.6. Evaluation criteria

In order to evaluate the effects of search methods on the negotiation performance of each bidding strategy, we use a set of six criteria to measure different aspects of negotiation performance. Our evaluation measures are used to demonstrate the quality of the achieved agreements by each agent and the efficiency of its negotiation strategy for gaining such agreements. To this end, we pay attention to the *individual* and joint utility (*social welfare*) obtained in negotiations. We have also focused on the extent to which the agent has been able to find a *optimal* and *fair* solution in its total negotiations. The mean *Pareto Distance* and mean *Nash Distance* have been chosen for these objectives. Moreover, two time measures (passed *time in seconds* and the *number of rounds*) have been used to evaluate the time efficiency of bidding strategies and examine how different search methods affect the required time to agreement. The time measures can give us an insight of how bid search methods can save time for the agent and help it reach earlier (and possibly better) agreements. The details of the evaluation criteria are presented in Appendix A.

4. Results and Discussion

In this section, we present the results of the experiments and analyze what has been obtained. We present the evaluation results in terms of the three experiment stages described in the previous section.

4.1. Experiment A

In this experiment, the implementation of search algorithms on two concession tactics (Boulware and linear Conceder) is tested using 6 baseline agents. We obtained the average values in the negotiation performance criteria for each of the 9 agents in negotiations with the 6 baseline strategies. The results of the experiment were organized according to two mentioned categories and the three negotiation domains which gave us 6 different tables. To avoid prolonging the content of the article, the detailed results of the experiment have been presented in Appendix B.

In order to have an estimate of how meta-heuristics affect the bidding strategies and to have a picture of the performance of each algorithm, the test results must be aggregated. To do this, in table 4 we summarize all the results and compare the search methods based on their successful performance in negotiation performance criteria. Table 4 shows the overall performance of search methods in the whole negotiations for both Boulware and Linear Conceder categories and in the total of three negotiation domains according to different criteria. Each column in table 4 is related to a negotiation performance measure and shows the number of successful performances for each agents using the corresponding search method. To better evaluate methods, we define successful performance for a search method in each measure as follows:
The number of times an agent with the search method was ranked from first to third in results tables.

Table 4
Overall performance of the search methods in experiment A according to successful performance criterion

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	2	3	0	6	3	1
BS	3	3	5	5	2	1
LS	4	3	6	2	3	3
VNS	1	2	0	0	0	1
SA	2	3	1	3	2	1
TA	2	2	0	2	3	3
TS	3	1	4	2	0	3
GA	3	3	1	0	4	0
ACO	0	0	0	3	4	5
Best agents	LS	ES,BS,LS, SA,GA	LS	ES	GA, ACO	ACO

For example, it can be seen that the LS method, was able for 4 times, to help an agent with any of bidding strategies to achieve a *utility* that was one of the top three places. The reason for choosing an evaluation criterion like this is to avoid bias in selecting a particular method, eliminating the effect of errors in the data and taking into account all the methods with really close performances for the evaluation process. The agent with the highest number of top rankings is selected as the successful agent in that criterion. The last line of Table 4 shows the top agents in each performance criterion.

According to Table 4, the simple local search (LS) is among the top methods in three criteria: utility, social welfare and time. This method is especially superior to others in two criteria, the obtained *utility* and the *time* to agreement and has been ranked first in absolute terms. It was even more successful in the average time to agreement (in seconds) than the binary search method, which is of order $o(\log n)$. Considering the fact that there is no need to sort the outcome space based on utility to apply the local search algorithm, the superiority of LS over BS in negotiation is very significant. Although the LS has been very successful in the actual time needed to get agreement, the two basic search methods, ES and BS has been the best methods in the number of rounds to reach agreement. This reveals interesting fact. Since the ES and BS methods spend more time on searching the outcome space and explore the whole search space, they better succeed in finding the right bids and therefore end the negotiations in less number of rounds.

Another interesting point from Table 4 is that the two population-based methods, GA and ACO outperform others in the Pareto distance criterion. Also, in the Nash distance criterion, the ACO algorithm has absolute superiority. This result shows that deployment of population-based algorithms and *particularly the methods based on swarm intelligence* can help to find optimal and fair agreements according to theoretical measures for negotiation in large domains. On the other hand, *single-solution* methods can be used to achieve more individual utility and social welfare.

4.2. Experiment B

According to the results of the first experiment, and considering the performance of search methods in different criteria, 6 top search methods were selected. The selected methods are: LS, SA, GA, ACO, ES, BS. After performing Experiment B, we have presented the results in accordance with the negotiation domains. Fig. 4, 5 and 6 show the utility of the found offer in comparison with the target utility for each of the 6 selected methods in the domains D1, D2, and D3, respectively. In these graphs, the target utility has been obtained using *Boulware* concession tactic.

At first glance, it is clear that the ACO algorithm finds the desired offer with a lot of fluctuations as long as the agent concedes strictly. Sometimes it even has a distance of more than 0.1 from the target utility, which does not show good performance for a search algorithm.

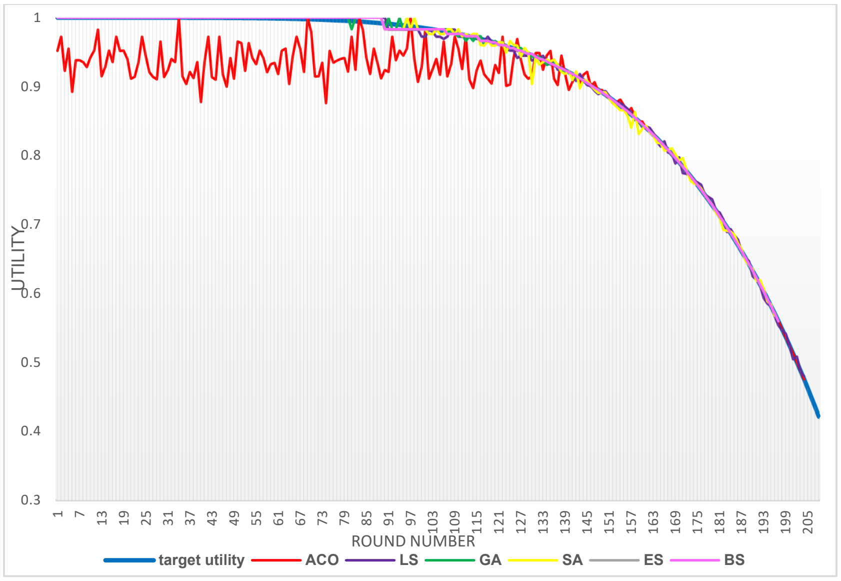

Fig. 4. The utility of the bids found by 6 search methods for domain D1

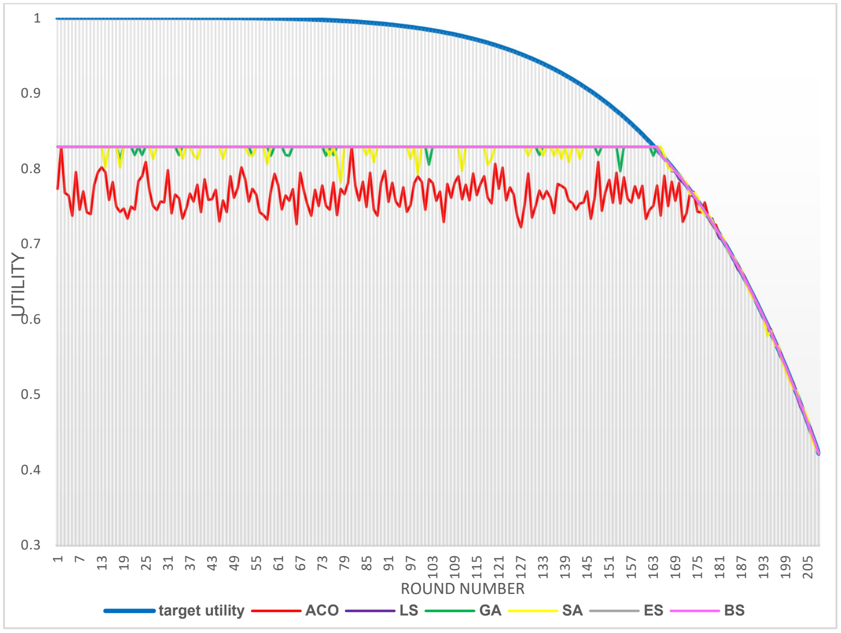

Fig. 5. The utility of the bids found by 6 search methods for domain D2

These fluctuations are due to the random nature of the ACO algorithm and the existence of a trade-off between the algorithm iterations and the quality of the solution found. The ACO algorithm requires a lot of computation resources to achieve convergence due to the use of ants collective behavior, pheromone matrix updates, and a large number of iterations. The rest of the methods show more or less the same accuracy, although, both SA and GA also show small fluctuations around the target utility. The three methods LS, BS and ES can accurately find the desired offers. Therefore, meta-heuristic methods such as SA, LS and GA are better alternatives to exhaustive and binary search methods as they require less time to find the right bid in the large outcome space.

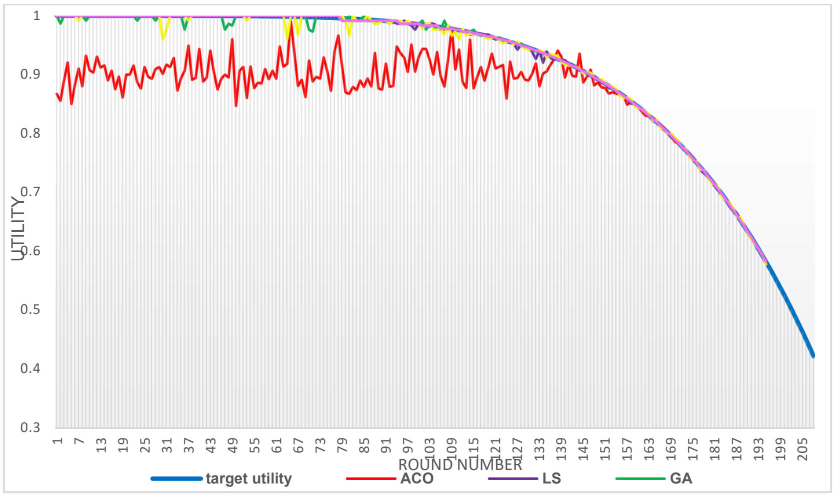

Fig. 6. The utility of the bids found by 6 search methods for domain D3

4.3. Experiment C

The results of the negotiation leagues for each group of agents with 6 selected search methods were organized according to negotiation performance criteria so that we could identify successful search methods for improving each criterion. Fig. 7 shows a comparison of search methods in terms of the average amount of utility obtained when combined with *Boulware* and *Linear Conceder* tactics. In the Boulware category, it can be seen that the three methods LS, ACO and SA gain more utility than BS does. The LS and ACO are always among the top three ranks. In the Linear Conceder category, given the compromises made by this tactic, most search methods operate at the same level. In particular, the three methods SA, ACO and GA show stable and good performance in all three domains. These three methods outperform ES and BS in domains D2. So, it can be concluded that the three meta-heuristic algorithms can be useful for Linear Conceder strategy. In a final review, we can say that the three methods LS, SA and ACO are successful methods in gaining utility.

Fig. 8 shows the amount of social welfare obtained for each search method in the two categories. For this criterion, SA and ACO provide good performance in a Boulware strategy, achieving social welfare equal to or more than BS and ES methods. In Linear Conceder, meta-heuristic methods do not outperform the two conventional search methods, but GA has performed better than other meta-heuristics. Fig. 9 provides a similar comparison for the amount of Nash distance. For a Boulware agent, the ES, with twice the first rank and one as a second rank, has had the best performance in finding offers with the shortest Nash distance. SA and LS are the next successful methods. In the conceder strategy, it is not possible to find a specific method that performs best overall. The GA method, which did not work well in a Boulware strategy, works better in this category, and it can be said that this method is concession-friendly, because in the case of linear conceding, it always works well (one of the first to third ranks), although not Great (according to Fig. 7 to 9). In general, the LS is an acceptable method for obtaining a good Nash distance or finding a fair agreement for both strategies. Fig. 10 also shows the Pareto distance obtained for different search methods. Surprisingly, the LS search method works extremely well in the area of Boulware tactic. But in Conceder strategy, it does not perform well in both D1 and D2.

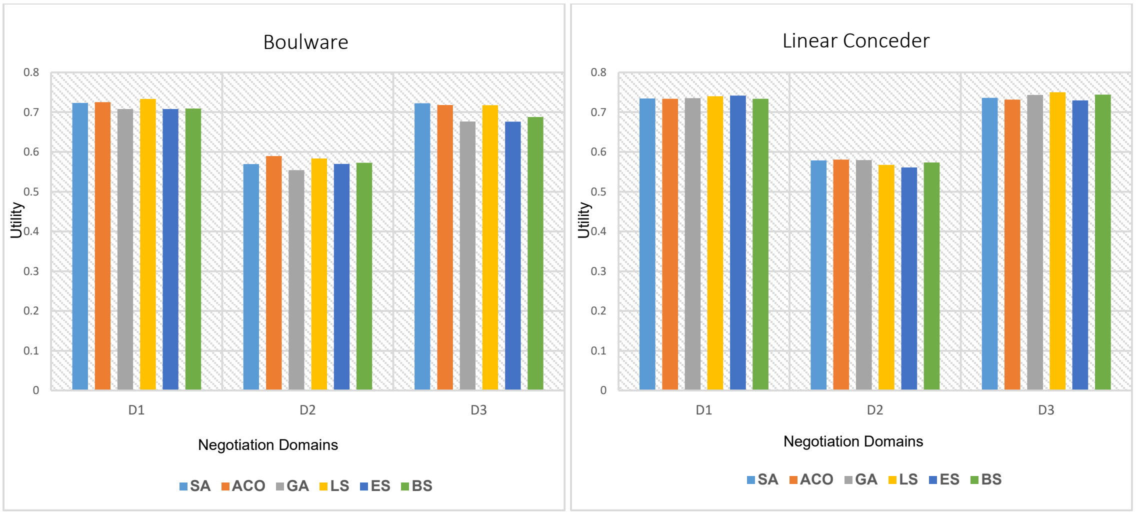

Fig. 7. The utility obtained with each search method in two tactic categories on three domains

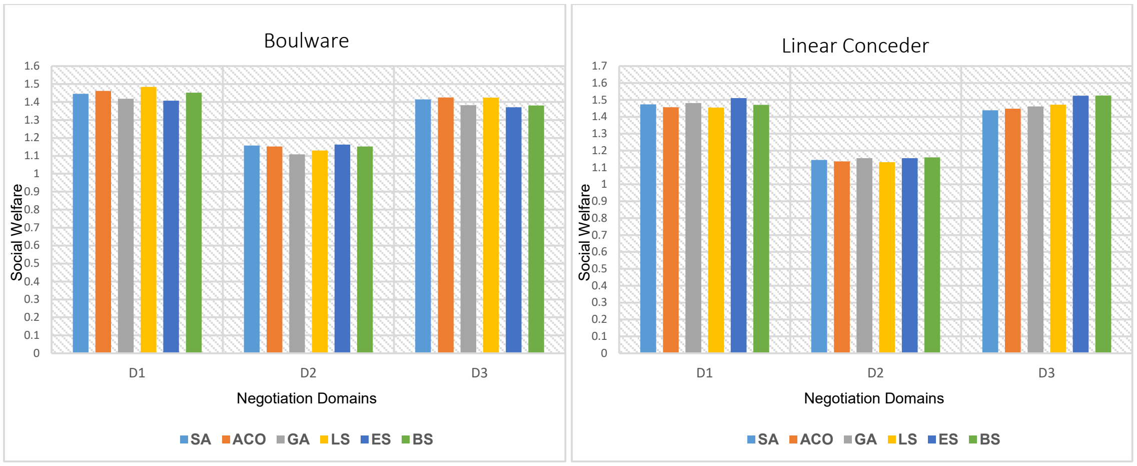

Fig. 8. The social welfare obtained with each search method in two tactic categories on three domains

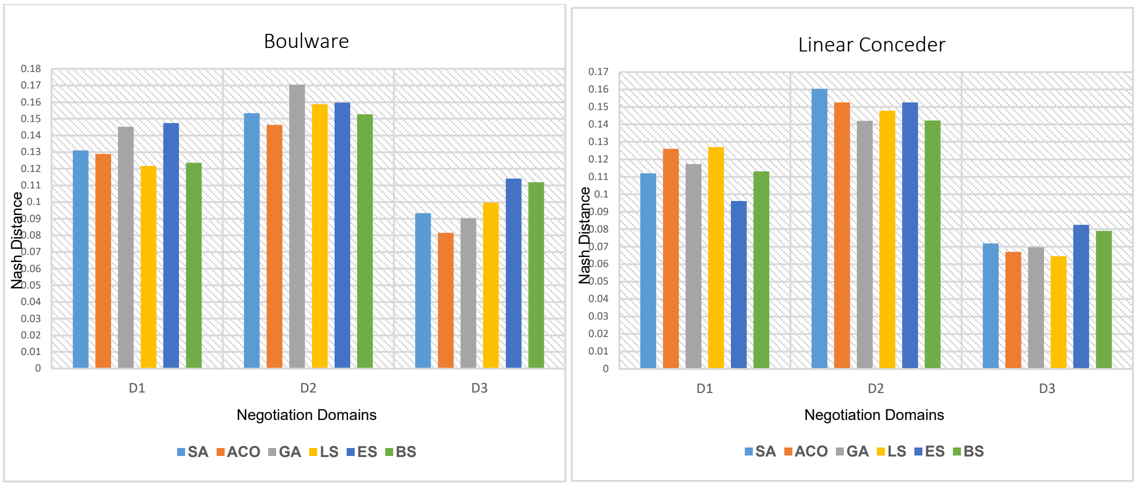

Fig. 9. The Nash distance obtained with each search method in two tactic categories on three domains

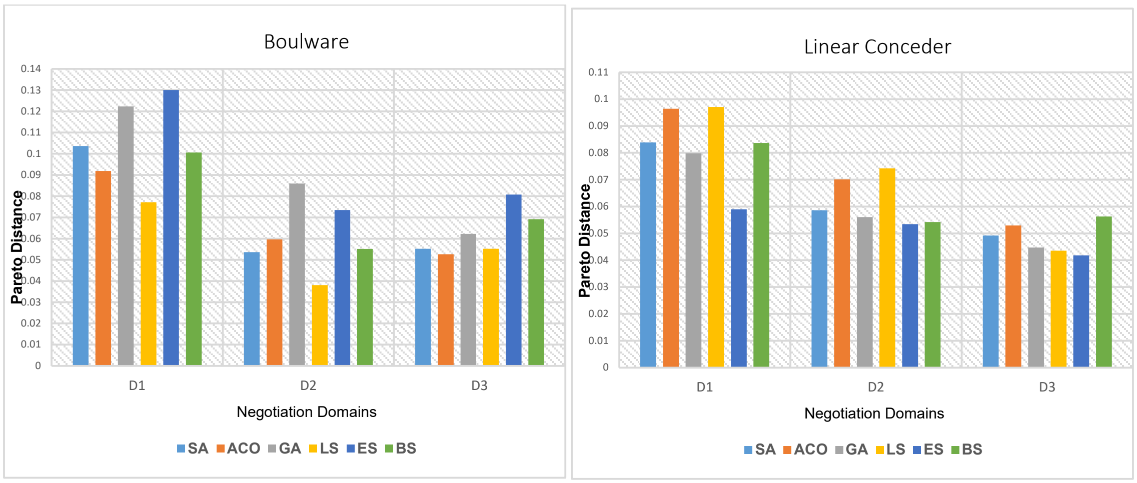

Fig. 10. The Pareto distance obtained with each search method in two tactic categories on three domains

In the D3 domain, which has a larger space, this method has shown better performance with the second rank. Interestingly, the ES method ranks first in all three domains for Conceder category. Therefore, to search for a fair bid in Linear Conceder mode, an exhaustive search in the outcome space will be successful, provided that enough computational resources are available. Furthermore, according to Fig. 9 and 10, it can be seen that the ACO method also shows good performance in the Boulware category. These findings are somewhat consistent with the results of Experiment A and it can be said that this method is successful to reach fair and optimal agreements.

Table 5
Average time to agreement for agents using each search method with the Boulware tactic

Search	Time (seconds)			Rounds		
Method	D1	D2	D3	D1	D2	D3
ES	33.3941	141.0511	311.0361	185	198.9	189
BS	15.1787	**38.7242**	75.892	183	198.8	189.1
LS	**13.3753**	40.0178	**70.5143**	**181.3**	199.8	187.2
SA	23.9903	50.7474	84.466	182.9	**198.6**	187.4
GA	19.5895	45.2389	77.9716	185	200.1	188
ACO	40.8386	73.0904	106.1944	182.8	198.8	**185.7**

Table 6
Average rounds to agreement for agents using each search method with the Conceder tactic

Search	Time (seconds)			Rounds		
Method	D1	D2	D3	D1	D2	D3
ES	11.928	74.1003	110.4051	**65.4**	103.8	69
BS	**5.7756**	**20.1358**	**25.0374**	70.3	**101.4**	**68.3**
LS	6.0018	21.6088	26.9764	70.6	107.3	70.9
SA	8.6781	26.3911	31.9584	67.3	105.7	74.6
GA	8.6515	23.5215	27.1082	*67.2*	104.3	70
ACO	15.6032	38.6677	38.681	69.6	105.5	71.9

In Tables 5 and 6, we have compared the time taken to reach an agreement for agents using different search methods on different domains. The best method in each column has been shown with bold font, while the second ranked method has been denoted using underline format. It is observable that among meta-heuristic algorithms, LS has the best performance. In the Boulware case, this method has recorded the minimum time required to reach an agreement in two domains (D1, D3). In the linear concession tactic, the BS method works best in all three domains, but the LS method is very close to that. As mentioned, the binary search method requires sorting the search space with the complexity of $O(\text{logn})$ or $o(n^2)$ to execute properly. It can be concluded that since LS does not require this time to sort the search space before the start of the negotiation, it is a better method than BS in terms of time complexity. In order to make a complete conclusion about the impact of each search method on each criterion of negotiation performance, this time we calculate the total successful performance of each search method for each criterion in both concession categories and on negotiation domains. In this case, we consider the successful performance as *the emergence of a search method within the top two methods in each negotiation performance measure*. Where the difference between the second and third ranks is very small, we also consider the method in the third rank. The result of this selection is presented in Table 7.

Table 7
Final evaluation of search methods in Experiment 3 according to their successful performance in each negotiation performance criterion

Search Method	Utility	Social Welfare	Distance to Pareto	Distance to Nash	time (s)	# bids (rounds)
ES	1	4	3	1	0	4
BS	3	3	1	2	6	4
LS	5	2	4	3	6	2
SA	4	4	2	2	0	4
GA	3	2	0	2	0	1
ACO	4	4	2	4	0	3
Recommended Method	**LS**, SA, ACO	ES, **SA**	**LS**	**ACO**, LS	**LS**, BS	**SA**, ES, BS

The last row denotes the successful and recommended methods to be used in order to improve each criterion. The method that is superior has been displayed in bold format.

By studying Table 7, it is found that the LS method is the best method among the other search methods in the three criteria of *individual utility*, *Pareto distance* and *time* required to reach an agreement. In addition, this method ranks second in the Nash distance and is acceptable to use. Therefore, it can be said that the local search method is the best method to use in a large outcome space, as it can find the desired offer faster while improving the quality of negotiation agreement. The simplicity of the local search method, the use of the hill climbing algorithm to reach the nearest peak, fewer number of iterations and the lack of complex calculations have made LS an acceptable method for searching bids in large spaces. It is also beneficial to pay attention to the performance of the SA method, because this method has the best performance in terms of social welfare. Also, in terms of individual utility, it is ranked second after LS. We can conclude that single solution-based meta-heuristic methods offer acceptable performance for searching in a large space. Population-based methods did not perform significantly in this study. This can be attributed to the complexity of the algorithms, the high computations per iteration, the time required for the algorithm to converge, and the need to tune many parameters, which reduces the efficiency of the method compared to single-solution methods.

5. Conclusion

In this paper, we presented a well-defined methodology with rigorous experimental evaluations to investigate the potential of using meta-heuristic methods for optimizing bid search in large outcome spaces. In order to effectively evaluate the benefits and impacts of metaheuristics on the bid search, we established two baseline bidding strategies using two important time-dependent tactics and implemented some well-known meta-heuristic algorithms in the search module of those baseline strategies. Then, we evaluated the negotiation performance of various agents that used metaheuristic-based search methods as well as those with traditional bid search methods. The results of the evaluations demonstrated that single solution-based metaheuristics, such as simulated annealing and hill climbing can generally perform a better bid search, bringing more utility and social welfare for the negotiating agents than population-based metaheuristics. Hill climbing, which is a basic metaheuristic with steepest decent algorithm is the most successful search method among all the metaheuristics tested in this study. This method outperforms others in three performance criteria: individual utility, Pareto distance and time, whilst its performance is acceptable in the Nash distance criteria. According to the experiments, we can recommend to use ACO, a swarm-based intelligence to improve Nash distance in negotiations. Our findings demonstrates that although the complex metaheuristic algorithms can be helpful to solve large optimization problems, in the context of automated negotiation, simpler metaheuristics with fewer number of iterations and the lack of complex calculations can provide more acceptable performance for searching bids in large spaces.

The continuation of this research can be suggested in two different directions. First, the metaheuristic algorithms can be implemented on other types of bidding strategies that might not present a baseline tactic to find out about the effect of metaheuristics on a broader set of negotiation strategies. It will be helpful to implement metaheuristics on behavior-based tactics to see how they can perform the bid search using the target utility that is obtained based on opponent's behavior. Second, more metaheuristic methods, including the novel proposed methods in recent years, can be implemented in the bid search module of a negotiating agent. This way a better knowledge of the metaheuristics in different types will be gained. In addition, new experiments can be conducted to compare the performance of single solution-based and population-based metaheuristics and examine their effect on different negotiation performance criteria.

References

Agrawal, M. K., & Chari, K. (2009). Learning negotiation support systems in competitive negotiations: A study of negotiation behaviors and system impacts. *International Journal of Intelligent Information Technologies*, *5*(1), 1–23.

Amini, M., Fathian, M., & Ghazanfari, M. (2020). A boa-based adaptive strategy with multi-party perspective for automated multilateral negotiations. *Applied Intelligence*, 1–31.

Baarslag, T. (2016). *Exploring the Strategy Space of Negotiating Agents: A Framework for Bidding, Learning and Accepting in Automated Negotiation*. Springer.

Baarslag, T., Aydogan, R., Hindriks, K. V., Fujita, K., Ito, T., & Jonker, C. M. (2015). The automated negotiating agents competition, 2010–2015. *AI Magazine*, *36*(4), 115–118.

Baarslag, T., Hindriks, K., Hendrikx, M., Dirkzwager, A., & Jonker, C. (2014). Decoupling negotiating agents to explore the space of negotiation strategies. In *Novel insights in agent-based complex automated negotiation* (pp. 61–83).Springer.

Baarslag, T., Hindriks, K., & Jonker, C. (2014). Effective acceptance conditions in real-time automated negotiation. *Decision Support Systems*, *60*, 68–77.

Burke, E. K., Kendall, G., et al. (2014). *Search methodologies* (second ed.). Springer.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms*. MIT press.

Davis, L. (1991). Handbook of genetic algorithms.

De Jonge, D., & Sierra, C. (2016). Gangster: an automated negotiator applying genetic algorithms. In *Recent advances in agent-based complex automated negotiation* (pp. 225–234). Springer.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical computer science*, *344*(2-3), 243–278.

Dowsland, K. A. (2014). Introduction. In *Search methodologies* (pp. 1–17). Springer.

Dueck, G., & Scheuer, T. (1990). Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, *90*(1), 161–175.

El-Ashmawi, W. H., Abd Elminaam, D. S., Nabil, A. M., & Eldesouky, E. (2020). A chaotic owl search algorithm based bilateral negotiation model. *Ain Shams Engineering Journal*.

Faratin, P., Sierra, C., & Jennings, N. R. (1998). Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, *24*(3-4), 159–182.

Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2001). Optimal negotiation strategies for agents with incomplete information. In *International workshop on agent theories, architectures, and languages* (pp. 377–392).

Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2002). Multi-issue negotiation under time constraints. In *Proceedings of the first international joint conference on autonomous agents and multiagent systems: part 1* (pp. 143–150).

Fujita, K., Aydogan, R., Baarslag, T., Ito, T., & Jonker, C. (2016). The fifth automated negotiating agents competition (anac 2014). In *Recent advances in agent-based complex automated negotiation* (pp. 211–224). Springer.

Gendreau, M., Potvin, J.-Y., et al. (2019). *Handbook of metaheuristics* (third ed.). Springer.

Glover, F. (1997). Tabu search and adaptive memory programming—advances, applications and challenges. In *Interfaces in computer science and operations research* (pp. 1–75). Springer.

Hoos, H. H., & Stützle, T. (2004). *Stochastic local search: Foundations and applications*. Elsevier.

Jain, M., Maurya, S., Rani, A., & Singh, V. (2018). Owl search algorithm: a novel nature inspired heuristic paradigm for global optimization. *Journal of Intelligent & Fuzzy Systems*, *34*(3), 1573–1582.

Jennings, N. R., Faratin, P., Lomuscio, A. R., Parsons, S., Sierra, C., & Wooldridge, M. (2001). Automated negotiation: prospects, methods and challenges. *International Journal of Group Decision and Negotiation*, *10*(2), 199–215.

Kadono, Y. (2016). Agent yk: An efficient estimation of opponent's intention with stepped limited concessions. In *Recent advances in agent-based complex automated negotiation* (pp. 279–283). Springer.

Kennedy, J. (2006). Swarm intelligence. In *Handbook of nature-inspired and innovative computing* (pp. 187–219). Springer.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of icnn'95-international conference on neural networks* (Vol. 4, pp. 1942–1948).

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Knuth, D. E. (1998). *The art of computer programming: Volume 3: Sorting and searching*. Addison-Wesley Professional.

Marsa-Maestre, I., Klein, M., Jonker, C. M., & Aydogan, R. (2014). From problems to protocols: Towards a negotiation handbook. *Decision Support Systems*, *60*, 39–54.

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

Mitra, D., Romeo, F., & Sangiovanni-Vincentelli, A. (1985). Convergence and finite-time behavior of simulated annealing. In *1985 24th IEEE conference on decision and control* (pp. 761–767). IEEE.

Mladenovic, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & operations research*, *24*(11), 1097–1100.

Niimi, M., & Ito, T. (2016). AgentM. In *Recent advances in agent-based complex automated negotiation* (pp. 235–240). Springer.

Osborne, M. J. (2004). *An introduction to game theory* (Vol. 3, No. 3). New York: Oxford university press.

Raiffa, H. (2007). *Negotiation analysis: The science and art of collaborative decision making*. Harvard University Press.

Sato, M., & Ito, T. (2016). Whaleagent: Hardheaded strategy and conceder strategy based on the heuristics. In *Recent advances in agent-based complex automated negotiation* (pp. 273-278). Springer, Cham.

Talbi, E.-G. (2009). *Metaheuristics: from design to implementation* (Vol. 74). John Wiley & Sons.

Yang, X.-S. (2017). *Nature-inspired algorithms and applied optimization* (Vol. 744). Springer.

Appendix A. Evaluation criteria

In the following, we introduce our negotiation performance criteria:

a) *Mean Received Utility*, is the average utility value that the agent gains in all negotiation sessions. For agent A, the average individual utility in 'n' negotiation sessions is calculated from the following formula:

$$MRU_A = \frac{1}{n}\sum_{i=1}^{n} u_A^i \tag{A.1}$$

where u_A^i A is the utility that agent A obtains in i-th session.

b) *Mean Social Welfare*, is the average value of the total utility obtained by both parties in each of negotiation sessions in which the agent *A* participated. The higher this value, the more beneficial and satisfying a deal is for all parties. This measure is calculated by the following equation:

$$MSW_A = \frac{1}{n}\sum_{i=1}^{n} SW_A^i \tag{A.2}$$

in which SW_A^i is the sum of the utilities obtained by both agents in the i-th negotiation session between agent A and its opponent agent B:

$$SWA^i = u_A^i + u_B^i, \tag{A.3}$$

c) *Mean Pareto Distance*, shows the average distance of obtained agreements from the Pareto frontier in the utility space over all negotiation sessions. This criterion is used to measure how optimal an agreement is according to Pareto frontier. The mean Pareto distance on 'n' sessions is calculated as follows:

$$MPD_A = \frac{1}{n}\sum_{i=1}^{n} PD_A^i, \tag{A.4}$$

in which PD_A^i is the distance of the obtained agreement between agent A and its opponent from the Pareto frontier in the i-th session. If agent A is negotiating with its opponent B, then:

$$PD_A^i = \sqrt{(u_A^i - P_A^i)^2 + (u_B^i - P_B^i)^2} \tag{A.5}$$

u_A^i is the utility of the achieved agreement for agent A and u_B^i is the utility of the agreement for the opponent in i-th session. In addition, P_A^i is the utility of the nearest point on the Pareto frontier for agent A and P_B^i is the utility of such a point from the viewpoint of the A's opponent.

d*) Mean Nash distance*, is the mean distance between the obtained agreements and the Nash solution over all negotiation sessions. The Nash solution or Nash equilibrium is a popular solution in game theory to determine the optimal result of a non-cooperative game. In the negotiation utility space it shows the point located on the Pareto frontier in which the product of the both agents' utility is maximized. This is a criterion used to measure the fairness of an agreement. If Agent A is negotiating with k opponents in each session, then the average Nash distance on 'n' sessions is obtained based on the following equation:

$$MND_A = \frac{1}{n}\sum_{i=1}^{n} ND_A^i, \tag{A.6}$$

in which ND_A^i A is the distance of the obtained agreement by agent A and its opponent from Nash point in the i-th session. If agent A is negotiating with its opponent B, then:

$$ND_A^i = \sqrt{(u_A^i - N_A^i)^2 + (u_B^i - N_B^i)^2}. \tag{A.7}$$

In the above equation, u_A^i is the agreement utility for agent A and u_B^i is the agreement utility for B in i-th session. Similarly, N_A^i and N_B^i are the the utility of the Nash solution for agent A and its opponent B respectively.

E) *Avg. time of agreement*, is the average amount of time (in seconds) taken for an agent to reach an agreement:

$$AT_A = \frac{1}{m}\sum_{i=1}^{m} T_A^i, \tag{A.8}$$

in which T_A^i is the the number of seconds that it takes until agent A reaches an agreement with its opponent in i-th negotiation session and m is the total number of agreed negotiation sessions.

f) *Avg. rounds of agreement*, is the average number of rounds passed for an agent to reach an agreement. This measure is calculated as follows:

$$AR_A = \frac{1}{m}\sum_{i=1}^{m} R_A^i, \tag{A.9}$$

in which R_A^i is the the number of rounds passed until agent A reaches an agreement with it opponent in i-th negotiation session and m is the total number of agreed negotiation sessions.

Appendix B. The detailed results of Experiment A

Tables B.1 to B.6 show the average values obtained in the negotiation performance criteria for each of the 9 agents in the face of the 6 baseline strategies separated by the categories and negotiation domain. Tables B.1 and B.2 show the negotiation performance of 9 agents over the D1 domain with respect to each of the Boulware and Linear Conceder strategies. Tables B.3 and B.4 show the results of the same experiments for the domain D2. Also, Tables B.5 and B.6 display the results for the domain D3. In all tables B.1 to B.6, the best method is each criterion has been shown with bold font, while the second and third methods have been displayed using underline and italic forms.

Table B.1
Performance of 9 search methods on Boulware strategy for D1 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	**0.893440143**	1.491949322	21.8428	**93.98333333**	0.044900996	0.259745385
BS	**0.893440143**	1.491949322	14.15285	**93.98333333**	0.044900996	0.259745385
LS	0.89260988	*1.494338662*	**10.3695**	102	*0.043374113*	0.258332281
VNS	0.887712709	1.486342891	15.564	102.9166667	0.049443141	0.262657804
SA	0.892870382	0.892870382	13.91766667	101.5	0.043537743	0.263716802
TA	*0.892775479*	**1.507753297**	14.95041667	*100.5*	**0.033966513**	0.260455008
TS	0.886096294	1.491406401	10.76283333	103	0.047291551	*0.258485638*
GA	0.890536107	1.49685709	13.36425	101.9166667	0.042859655	0.259072051
ACO	0.856220031	1.474355408	26.08075	97.25	0.064378615	**0.229512632**

Table B.2
Performance of 9 search methods on Linear Conceder strategy for D1 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	0.759001414	**1.531849103**	13.36956667	62.43333333	**0.023427814**	0.228316146
BS	**0.76819584**	1.522545404	5.241333333	64.03333333	0.026925768	**0.227360859**
LS	*0.76233916*	1.511406683	**5.236**	*63.66666667*	0.039398603	*0.23503461*
VNS	0.751612253	1.495250001	8.837583333	64.08333333	0.05074906	0.248512657
SA	0.759086127	*1.514730104*	10.84558333	**61.91666667**	*0.037745414*	0.236417103
TA	0.749779276	1.503738912	10.9855	63.75	0.044271434	0.246663163
TS	0.750429526	1.496630121	*7.338833333*	63.83333333	0.049335464	0.247603312
GA	0.764385997	1.503603329	11.06266667	65.75	0.04193107	0.239641921
ACO	0.737498146	1.499491599	17.76016667	63.83333333	0.045255745	0.235528226

Table B.3
Performance of 9 search methods on Boulware strategy for D2 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	**0.752008323**	**1.233039793**	112.5367	109.1833333	**0.040994314**	0.239192617
BS	**0.752008323**	**1.233039793**	40.83321667	109.1833333	**0.040994314**	0.239192617
LS	0.74433815	1.208474528	**40.14575**	118	0.05894058	0.246049869
VNS	0.743616813	1.212024824	47.69533333	118.0833333	0.055909879	0.246818649
SA	0.736904024	*1.216147069*	50.27833333	*116.75*	0.054726619	0.236842143
TA	0.735344576	1.20700965	51.92083333	117.1666667	0.052249462	*0.237554908*
TS	*0.74466962*	1.205202793	*47.67333333*	**101.4416667**	0.062219335	0.246620308
GA	0.74600693	1.23054431	48.641	117.0833333	0.04255598	0.238170052
ACO	0.691148171	1.185873008	65.92666667	115.4166667	*0.051897209*	**0.189151703**

Table B.4
Performance of 9 search methods on Linear Conceder strategy for D2 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	0.605967596	1.211935193	91.31775	**83.58333333**	0.058319051	0.238786064
BS	0.605967596	1.211935193	**30.6472**	**83.58333333**	0.058319051	0.238786064
LS	**0.613481175**	1.222688887	30.95016667	84.91666667	*0.051130585*	*0.237142698*
VNS	0.610727296	1.213527785	36.83766667	86.16666667	0.056979218	0.241378802
SA	0.602557401	1.206789506	36.98333333	86.41666667	0.061529572	0.240714565
TA	*0.612013625*	*1.217989917*	37.709	*85.75*	0.05216	0.238048387
TS	0.605115587	1.215822105	*32.915*	85.91666667	0.056828305	0.235310393
GA	0.612697235	**1.224633044**	36.48366667	86.66666667	0.04496298	0.246976958
ACO	0.588141764	1.212235798	48.06275	85.83333333	**0.039110674**	**0.215195992**

Table B.5
Performance of 9 search methods on Boulware strategy for D3 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	0.877660622	1.463749721	317.5623667	**93.65**	0.095084149	0.250292556
BS	0.877660622	1.466006016	*118.4524333*	**93.65**	0.092966293	0.249313774
LS	0.918605551	1.496734859	**106.95575**	101.4166667	0.08898186	0.261759485
VNS	0.907117339	*1.512529204*	128.2225	100.5	0.082320274	*0.237152235*
SA	**0.925226097**	1.517237369	130.2331667	100.9166667	*0.082224273*	0.250313327
TA	0.888791	1.503318407	130.0770833	99.33333333	0.07436718	0.219093123
TS	*0.913443436*	**1.522916508**	114.8195	*98.66666667*	0.086540415	0.24066644
GA	0.899436668	1.489668749	125.4374167	101.6666667	0.087815761	0.249117686
ACO	0.829425492	1.440193139	157.50525	98.5	**0.065294104**	**0.194569004**

Table B.6
Performance of 9 search methods on Linear Conceder strategy for D3 domain

Method	Utility	Social welfare	time (s)	# bids (rounds)	Distance to Pareto	Distance to Nash
ES	0.780342146	**1.563929599**	201.3366167	**59.95**	0.08554493	0.225065238
BS	0.780342146	**1.563929599**	**67.5395**	**59.95**	0.08554493	0.225065238
LS	0.782228417	*1.536500572*	72.04725	61.83333333	0.076748749	0.219358425
VNS	**0.788737288**	1.542192938	75.24975	62.33333333	0.074384312	0.221481235
SA	0.77646131	1.52036247	78.86291667	61	0.07229921	0.226851235
TA	0.769577561	1.521601537	76.14916667	62.83333333	*0.066255056*	**0.211304963**
TS	*0.78178425*	1.536145995	*73.09841667*	62	0.069191542	*0.21899913*
GA	0.770468382	1.518972832	77.21491667	62.08333333	0.062871828	0.22145027
ACO	0.766464776	1.510479601	82.95283333	*61.08333333*	**0.054690283**	0.212797446