Electronic Commerce Research and Applications 19 (2016) 56–69

Contents lists available at ScienceDirect

Electronic Commerce Research and Applications

journal homepage: www.elsevier.com/locate/ecra

Automated concurrent negotiations: An Artificial Bee Colony approach

Kostas Kolomvatsos [a,*], Kyriaki Panagidi [a], Ioannis Neokosmidis [b], Dimitris Varoutas [a], Stathes Hadjiefthymiades [a]

[a] Department of Informatics and Telecommunications, National Kapodistrian University of Athens, Greece
[b] Incites Consulting SARL, 130, Route d' Arlon, Strassen, Luxembourg

ARTICLE INFO

Article history:
Received 23 December 2015
Received in revised form 26 August 2016
Accepted 3 September 2016
Available online 9 September 2016

Keyword:
Concurrent negotiations

ABSTRACT

In *Electronic Marketplaces* (EMs), a number of unknown entities can interact to conclude purchase actions. Interactions are, usually, between buyers and sellers. Both groups of entities (e.g., buyers, sellers) aim to acquire items in the most profitable price. The discussed interactions are realized in the form of negotiations over a number of items characteristics. In this paper, we focus on the buyer side and deal with automated multi-issue concurrent negotiations. Such negotiations are between buyers and multiple sellers having in their property specific items. Each buyer negotiates with a number of sellers trying to achieve the most profitable value for a number of items' characteristics. We propose an optimization model for achieving the maximum possible utility. Our method adopts the principles of the *Artificial Bee Colony* (ABC) algorithm that offers a number of advantages compared to other *Swarm Intelligence* (SI) methods (e.g., *Particle Swarm Optimization – PSO*). The buyer, based on a number of threads, tries to find the optimal agreement when negotiating with a group of sellers. Every agreement, realized with a specific seller, results a utility for the buyer concluded over a weighted scheme on the items characteristics. Each thread adopts a weights adaptation model for optimizing the utility. A set of experiments reveal the strengths and weaknesses of the proposed model. We also report on a comparison assessment between the proposed method and other efforts found in the respective literature.

1. Introduction

Electronic Marketplaces (EMs) provide virtual places where unknown entities interact each other for exchanging items. In EMs, we can identify three types of users: the *buyers*, the *sellers*, and the *middle entities*. Buyers aim to purchase items while sellers have a number of items in their property and try to sell them in the most profitable price. Middle entities are mainly used for administration purposes (e.g., payments, security issues, etc). *Intelligent Agents* (IAs) could represent the discussed entities in EMs. IAs can undertake the responsibility of buying or selling items in an automated way. IAs can satisfy the constraints defined by their owners while they can, possibly, learn users' preferences, thus, increasing their performance.

Usually, in EMs, IAs interact each other, to conclude the exchange of items, in the form of negotiations. *Negotiation* is the process where unknown entities try to agree upon the exchange of specific items for specific returns (Raiffa, 1982). Negotiation is a decentralized decision making process undertaken to conclude an agreement that satisfies the requirements of two or more parties. Negotiations involve a number of offers exchanged between IAs (i.e., buyers and sellers) with the final aim of the purchase of items. In the respective literature, one can identify *bilateral* (one-to-one) or *one-to-many* negotiations. In the first case, a buyer negotiates with a seller. In the latter case, a buyer can negotiate, concurrently, with a number of sellers. Additionally, the negotiation could involve a single (e.g., price) or multiple issues (e.g., price, delivery time, quality, etc) for each item. In negotiations, buyers and sellers have conflicting goals. In a single issue negotiation, the buyer wants to purchase an item in the minimum possible price while the seller wants to sell the item in the highest possible price. Similar behavior is adopted in multi-issue negotiations, however, the type of a specific issue defines the attitude of the buyer/seller. For instance, the buyer wants to have an item in the minimum delivery time that it is not so important in the seller side.

In EMs, an item can be provided by a number of sellers, however, with different characteristics (e.g., price, delivery time). Every buyer should decide from which seller it is going to buy the desired product. This can be achieved through the adoption of *concurrent*

* Corresponding author.
E-mail addresses: kostasks@di.uoa.gr (K. Kolomvatsos), k.panagidh@di.uoa.gr (K. Panagidi), D.Varoutas@di.uoa.gr (I. Neokosmidis), shadj@di.uoa.gr (D. Varoutas), i.neokosmidis@incites.eu (S. Hadjiefthymiades).

http://dx.doi.org/10.1016/j.elerap.2016.09.002
1567-4223/

negotiations to increase the performance and conclude the best possible agreement. In such a setting, the buyer concurrently negotiates with a number of sellers and accordingly chooses the best agreement among all. The buyer utilizes a number of *threads* in order to negotiate with every seller. In such cases, a *coordination module* (CM) is necessary that can, probably, define the strategy for each thread. To the best of our knowledge, the majority of the research efforts found in the literature in the concurrent negotiations domain, adopt a CM that, probably, could be not efficient when applied in real scenarios. The following list reports on the drawbacks of the CM responsible to 'guide' threads during negotiations:

- *The CM consists of a single point of failure*. Any malfunction in the CM will affect the outcomes of the negotiations as threads cannot get instructions and adopt specific strategies;
- *A centralized scheme* (*i.e., the CM*) *requires an increased number of messages*. As the CM should instruct threads during negotiation, a significant number of messages should be exchanged between the CM and threads. This could cause bottlenecks and demand for increased resources in the CM while, in parallel, it increases the complexity in the CM's decision making mechanism (related to the strategies that each thread should follow);
- *The CM should be aware, in advance, for the appropriate strategies of any type of seller*. The centralized scheme demands for a complex strategy definition model that takes into consideration any potential aspect of the sellers' behavior. Any adaptation scheme during the negotiation increases the load of the CM and limits its performance.

In this paper, we examine the case where buyers negotiate concurrently with a number of sellers. We refer to a multi-issue negotiation and assume absolutely no knowledge on the entities' characteristics. Such knowledge involves deadlines, reservation prices, etc. *Reservation price* is the acceptable upper/lower limit of price for the buyer/seller. We try to increase the performance, in the buyer side, by adopting a methodology that makes unnecessary the use of a CM. The solution involves a number of self-adapting threads that adopt *Swarm Intelligence* (SI) (Engelbrecht, 2007). Through the adoption of SI, threads can independently readjust their strategies aligned with the negotiation information and the outcome in other threads. As no CM is necessary, the exchanged messages are minimized and, thus, the buyer saves time and computational resources. When an agreement is achieved by a specific thread, the remaining threads can readjust their strategy, if needed, to force the respective sellers to accept lower prices. After an agreement, each thread provides feedback to the seller for the final agreement after a specific time interval that enables the remaining threads to achieve lower prices. We propose a novel approach based on the known *Artificial Bee Colony* (ABC) optimization method (Karaboga, 2005). ABC could enhance the performance of our solution. ABC can better 'escape' from local minima compared to other SI techniques like *Particle Swarm Optimization* (PSO). In the examined negotiations, IAs try to achieve the best agreement, e.g., the agreement with the highest utility. The utility can be defined in many ways. For instance, it can represent the amount of money that the buyer saves (compared to an upper value) or the seller gains (compared to the item cost).

Example applications involve typical E-commerce models, Grid Computing and Cloud. For instance, a concurrent negotiation mechanism can be adopted for modeling the parallel negotiation activities between a broker agent and multiple groups of provider agents in Cloud (Sim, 2013). The aim is to establish multiple SLAs for a collection of resources. The discussed model consists of a module that manages the parallel negotiation activities for acquiring different types of Cloud resources in different resource markets. In each Cloud resource market, a broker agent establishes an SLA by negotiating simultaneously with multiple provider agents. The SLA is realized with the most profitable market as derived by the outcomes of successful negotiations. At each negotiation round, IAs determine whether to accept the proposed offer or to break an existing contract and get a new one paying a penalty fee. In this example, the broker agent plays the role of the buyer in our model while the provider agents play the role of sellers. In addition, our model can easily be combined with already defined payment systems. For instance, the proposed framework could be part of the transactions in a block chain wallet.[1] Users can create transactions and include a concurrent negotiation scheme to conclude purchases. Hence, our framework could be combined with the payment platform to derive a fully automated purchase framework that will facilitate end users to perform purchase actions. The envisioned threads after concluding a successful transaction with a specific seller could realize the payment adopting digital coins like bit coins. This automated process will increase the performance of the systems as the whole chain from the user request to the payment and the final conclusion of the transaction will be fully automated, thus, increasing the throughput.

The rest of the paper is organized as follows. Section 2 presents the related work in the discussed domain while Section 3 describes our scenario. We present our approach for concurrent multi-issue negotiations. In Section 4, we describe our proposed algorithm and in Section 5, we give specific experimental results. We compare our model with reference models found in the literature. Finally, in Section 6, we conclude our paper by presenting some future extensions.

2. Related work

Over the past years, a lot of work has been performed in automated negotiations. Many researchers have proposed models dealing with bilateral or concurrent negotiations (An et al., 2006; Chen and Huang, 2009; Da-Jun and Liang-Xian, 2002; Faratin et al., 1998; Fatima et al., 2005; Sun et al., 2007). We can group these efforts as follows: (i) approaches based on *Game Theory* (GT) (e.g., bargaining); (ii) approaches based on *Machine Learning* (ML) (e.g., learning or adapting on the opponent's strategy); (iii) approaches based on *Fuzzy Logic* (FL); (iv) approaches based on heuristic decision functions.

2.1. Negotiation models and strategies

Important role to negotiations plays the strategies of the entities as well as the interaction protocol. Usually, a specific *deadline* is set for each entity. The time for which the entities participate in a negotiation is important as it is used to exercise pressure on the entities. Finite horizon negotiations involve the exchange of alternating offers for a number of rounds (Stahl, 1972). However, one can find approaches where infinite horizon is considered (Stahl, 1972). Negotiations are realized over a *single* or *multiple issues*. The majority of the research efforts deal with single issue negotiations. Multi-issue negotiations are widely studied in the past (Fatima et al., 2005; Lau, 2005; Robu et al., 2005). Such negotiations can be further improved by incorporating heuristics (Jonker et al., 2004). IAs can utilize the history of the opponent's offers to predict her preferences or genetic algorithms for choosing the appropriate line of actions at every negotiation round (Lau, 2005). Multi-IAs concurrent negotiations involve a number of IAs and the proposed models deal with the behavior of the group (Türkay and Koray, 2012; Wu et al., 2009).

[1] https://www.blockchain.com/.

GT models usually study the bargaining game (Chatterjee and Samuelson, 1988; Crampton, 1984; Fudenberg et al., 1987; Jazayeriy et al., 2011; Shandholm and Vulkan, 1999). In many efforts, the authors assume knowledge of entities' characteristics or their distributions. For instance, reservation prices could be common knowledge (Crampton, 1984). Other parameters like the deadline, strategies or the type of entities could be also common knowledge (Chatterjee and Samuelson, 1988; Stahl, 1972). At every round of the negotiation, entities make an offer to the opponent and the latter entity has the opportunity to accept or reject it and make a counter offer. Each entity has a specific strategy. The strategy affects the offers or the response to the counter offers. Based on the strategies, the market equilibrium can be easily defined and analyzed. However, determining the equilibrium assumes that entities are rational and remain at the equilibrium path during negotiation. Unfortunately, entities can have outside options or their behavior could be restricted (Fudenberg et al., 1987). For instance, a seller could face an infinite number of buyers and have the opportunity to leave the negotiation at any time while buyers could not be capable of defining their offers but only to accept or reject the incoming proposals. Hard and soft strategies are already defined in the respective literature to depict the potential of an entity to be more strict or relaxed when sending offers (Chatterjee and Samuelson, 1988). Prior probabilities for reservation prices could be common knowledge or uncertainty could be present on both sides (Crampton, 1984). Time and information affect the rational behavior of IAs when commitment is not possible. The players should exchange some private information before an agreement is concluded. Pareto optimality is imperative for negotiation algorithms to lead to an efficient solution (Jazayeriy et al., 2011). The generation of a Pareto-optimal offer requires information about the opponent's importance weights.

In addition, a number of efforts deal with the definition of the interaction protocol and functions adopted to generate the entities' offers (Faratin et al., 1998; Fatima et al., 2005). Formal models and a set of tactics are proposed. Specific metrics are adopted to reveal the performance of the protocols like: i) the intrinsic benefit of the agent, ii) the cost, and, iii) the performance of the intrinsic utility relative to a complete knowledge interaction. The optimal strategy of entities is studied w.r.t. an item pricing scheme. Three types of functions are defined: linear (over time), boulware (the entity reaches its final offer slowly) and conceder (the entity reaches its final offer quickly). Finally, comparisons between concurrent and sequential interactions indicate that the concurrent model outperforms the sequential (Nguyen and Jennings, 2003a).

2.2. Uncertainty management in negotiation models

FL is the right tool for handling uncertainty that is inherent in dynamic environments. The negotiation strategies could be presented as fuzzy rules and simple heuristics could be employed to learn the preferences of the opponent (Cheng et al., 2005). Heuristics (Kolomvatsos et al., 2008b) and FL (Kolomvatsos et al., 2008a, 2015) could be also adopted to specify some of the negotiation parameters like the deadline. A set of fuzzy rules (Kolomvatsos et al., 2008a) or fuzzy constraints (Luo et al., 2003) could be defined according to experts' knowledge to determine a fair solution for all parties and, thus, several options that satisfy them are identified. In addition, fuzzy values on both sides could be adopted to incorporate the uncertainty in the definition of the entities' offers (Raeesy et al., 2007). Several ML models have been proposed for predicting the opponent's characteristics (Zuo and Sun, 2009). Genetic algorithms provide an efficient methodology for the prediction and learning of the opponent strategy (Gerding and van Bragt, 2003; Oliver, 1997). Additionally, the use of Bayesian models (Bui et al., 1995; Zuo and Sun, 2009; Leu et al., 2015) for learning the opponent's behavior increases the efficiency of the proposed systems. The approximate predictions are based on opponent's historical offers. The proposed approaches also incorporate a counter-offer definition algorithms which is capable of trading issues effectively based on the predicted preference of the opponent. By integrating the influences of the environment in a negotiation, negotiators can be effectively adapted to any change. An example ML algorithm adopted for learning the opponent's behavior is reinforcement learning (Zeng and Sycara, 1998). Reinforcement learning helps the entities to decide if their offers will be accepted. The aforementioned solutions have specific drawbacks when adopted in negotiations. Learning mechanisms aim to discover the optimal strategy as the response to the opponent's move and not to provide an efficient generic decision making mechanism. ML models require increased computational effort and the use of training samples while Bayesian learning requires the knowledge of a priori probability on the opponent type.

In addition, the application of prospect theory offers an efficient model for the preferences of negotiators modeled using S-shape value functions (Shyur and Shih, 2015). A unified agent incorporating three different types of concession tactics is developed while a function of simulation mimicking the process of the negotiation is provided. Discrete wavelet transformation and non-linear regression with Gaussian processes are adopted to model agents' opponents in real-time (Chen and Weiss, 2015). In this context, utility expectations are adaptively adjusted during negotiation. Adaptive probabilistic behavioral learning mechanisms for managing the opponent having unpredictable random behaviors are also proposed (Rajavela and Thangarathanam, 2016). To effectively learn the opponent's behavior over several stages of a negotiation process, a behavioral inference engine analyzes the sequence of negotiation offers. By modeling the negotiation process as the multi-stage Markov decision problem, appropriate counter-offer behavioral tactics generation based on the adaptive probabilistic decision taken over the corresponding negotiation stage are suggested.

2.3. Negotiations in grid computing

In Grid, brokers are adopted to handle the applications for resources (Haji et al., 2005). The resources that are capable of handling submitted jobs are differentiated by the broker. Considering that other users potentially require the same resource, probes are used to keep a vision of any changes, securing resources before submission and binding tasks to the resource. Users can be aware about concurrent negotiations that conflict one another through a *conflict-Aware Negotiation Protocol* for allocating Cloud resources and services (Netto, 2011). This approach limits the chances of having collisions with other users and facilitates users' decisions on the resources they want to allocate. The adopted negotiation protocol, usually, involves offers and counter-offers (Venugopal et al., 2008). According to this approach, a counter offer is generated as a response to an offer by modifying some of its terms through the alternate offers mechanism. Offers are generated by negotiation strategies aiming at long term allocations. For instance, the basic negotiation strategy is modeled using a polynomial function allowing clients to concede faster at the beginning of negotiation as compared to an exponential function (Haberland et al., 2015b). The choice among an infinite number of tactics is determined through sophisticated mechanisms taking into account task-specific characteristics and dynamics of Grid. Finally, adaptation on the adopted strategies provides more efficiency into the proposed negotiation mechanisms. Fuzzy control mechanisms can easily handle the uncertainty present in such dynamic environments (Haberland et al., 2015a). Estimates regarding the speed and the direction of the change in availability are performed by exploiting the information kept by the resource allocator. Hence, the

proposed solutions allow strategies adjustments according to resource availability changes during negotiation.

2.4. Concurrent negotiations

The focus of our model in on *automated concurrent negotiations*. Concurrent interactions, usually, have the form of one-to-many negotiations between a buyer and multiple sellers (Sun et al., 2007). The buyer is restricted to wait until the reception of offers from all the threads before generating the next offer. Furthermore, every seller can easily join and leave the negotiation dynamically. A set of coordination schemes try to alleviate the coordination activities in the buyer side (Rahwan et al., 2002; Nguyen and Jennings, 2003b). Some examples are: *Desperate*, *Patient*, and *Optimized Patient* strategies. Adaptive mechanisms have been proposed to increase the efficiency of the concurrent negotiation models (Narayanan and Jennings, 2005). Adaptation techniques provide a mechanism that changes the parameters of the alternative offers scheme (Narayanan and Jennings, 2005). In such cases, an adaptive model could adopt *Artificial Neural Networks* (Oprea, 2002). In a concurrent negotiation scheme, the CM plays the most significant role. The CM is responsible for choosing the appropriate strategy to be applied by each thread. It receives the status of each thread and decides the strategy based on the parameters of each interaction. Specific strategies could be adopted by the CM (Rahwan et al., 2002), e.g., *Fixed-Waiting-Time-Based* and *Fixed-Waiting-Ratio-Based* (An et al., 2006). Negotiation is conducted in continuous time and IAs rely on the discussed strategies to issue an offer. Experimental results suggest that the proposed mechanism achieves more favorable outcomes than the general, one-to-many methodologies. The proposed approach is based on the combination of a number of ad hoc heuristics involving a large set of parameters. Moreover, the CM can have a view on the overall negotiation process and manage the information related to the opponents. Based on this information, the CM could decide the best possible time to stop the negotiation and calculate the utility at that time (Williams et al., 2012). An in-depth analysis and the key insights accompanied by useful conclusions on the future extensions in the field are presented in the 2nd International Automated Negotiating Agents Competition (ANAC 2011) Baarslag et al., 2011. The participants analyze the strategies used in negotiation scenarios and techniques utilized by the teams. In particular, they show that the most adaptive negotiation strategies are not necessarily the ones that win the competition. Finally, recent efforts in the field involve the use of the known *Particle Swarm Optimization* (PSO) algorithm in concurrent negotiations (Kolomvatsos and Hadjiefthymiades, 2014; Panagidi et al., 2014). Threads try to be self-organized in order to reach to the best agreement.

2.5. Our contribution

In this paper, we study concurrent negotiations of a buyer with a number of sellers. The buyer decision process is based on a number of issues. The buyer utilizes a number of threads that negotiate with specific sellers. Each thread exchanges offers with a seller concerning a number of issues. We assume absolutely no knowledge on entities' characteristics. In contrast to other research efforts, like (Nguyen and Jennings, 2004; Rahwan et al., 2002; Sun et al., 2007), we do not need any CM to specify the strategy for each thread. Hence, the buyer requires less resources for the negotiation process. The CM consists of a conceptual single point of failure. If the CM fails to provide the necessary instructions to threads, the buyer cannot conclude the negotiations. Complex calculations in the co CM or messaging malfunctions could cause unnecessary delays. In the remaining efforts, the setting is a centralized approach. Many messages are required for the communication between the CM and threads while, in our case, the number of messages is minimized. In our model, even if messages are lost, threads could continue negotiations and reach the final result. Let us denote with N the number of threads and with T_b the deadline of the buyer. The centralized approach is of time complexity $O(N{\cdot}T_b)$ (for messaging) while our model is of time complexity $O(N)$ (for messaging). For example, if no agreement is concluded till T_b, the centralized approach needs $2{\cdot}N{\cdot}T_b$ messages to be exchanged between the CM and threads. In our case, the number of messages is zero. The differences of our work to those found in the literature are: a) each thread acts independently adapted to each seller strategy, b) we adopt SI for adjusting each thread's strategy to reach the optimal agreement. We choose to adopt the known ABC model instead of adopting the PSO methodology, like (Kolomvatsos and Hadjiefthymiades, 2014; Panagidi et al., 2013, 2014), to compare these systems and reveal their advantages and disadvantages in a concurrent negotiation setting. Recent research efforts reveal the increased performance of the ABC method (compared to the PSO model) in other research fields (e.g., clustering) (Karaboga and Ozturk, 2011). The ABC algorithm (Karaboga, 2005) was proposed as an optimization method of multivariate continuous objective functions. It belongs to the wider family of swarm optimization algorithms that are based on SI. These kinds of algorithms model the collective behavior of self-organized interacting swarms. It has been shown that ABC performance is comparable to other population-based methods (Goldberg, 1989; Karaboga and Akay, 2009) and has been used in several problems (Pan et al., 2011; Szeto et al., 2011) due to its simplicity and ease of implementation. The strongest advantage of ABC algorithm is its independency on the initial values of the examined variables.

3. Multi-issue concurrent negotiations

3.1. Negotiation setting

As mentioned, a set of threads undertakes the responsibility of negotiating with a specific seller. Let us focus on a single interaction between a thread and a seller. We model this interaction through a *finite-horizon multi-issue negotiation under incomplete knowledge*. The negotiation involves a number of alternating offers. We take into consideration the following assumptions:

- If one of the deadlines expires and no agreement is concluded, the negotiation ends with a conflict and zero utility for both entities;
- At each round of the process, the entities propose a set of values (package offer) for the examined issues (item characteristics). If the opponent accepts the offer, the negotiation ends with an agreement and positive utility for both;
- Each entity has absolutely no knowledge on the characteristics of its opponent;
- Both entities reason at every round and decide whether to continue the negotiation or not by rejecting the current offer (anticipating a higher utility at a later stage);
- The seller starts first and the thread follows, if the proposed offer is rejected. If an entity is not satisfied with the proposed offer, it has the right to reject it and issue a counter offer.

In Fig. 1, we can see the architecture of our model. At every round, each thread sends/receives an offer (a bundle of values) to the corresponding seller. If an agreement is true in a specific thread then the agreement message is sent to the rest of them (the message is received by every thread that currently participates in an active negotiation). We consider that the remaining threads

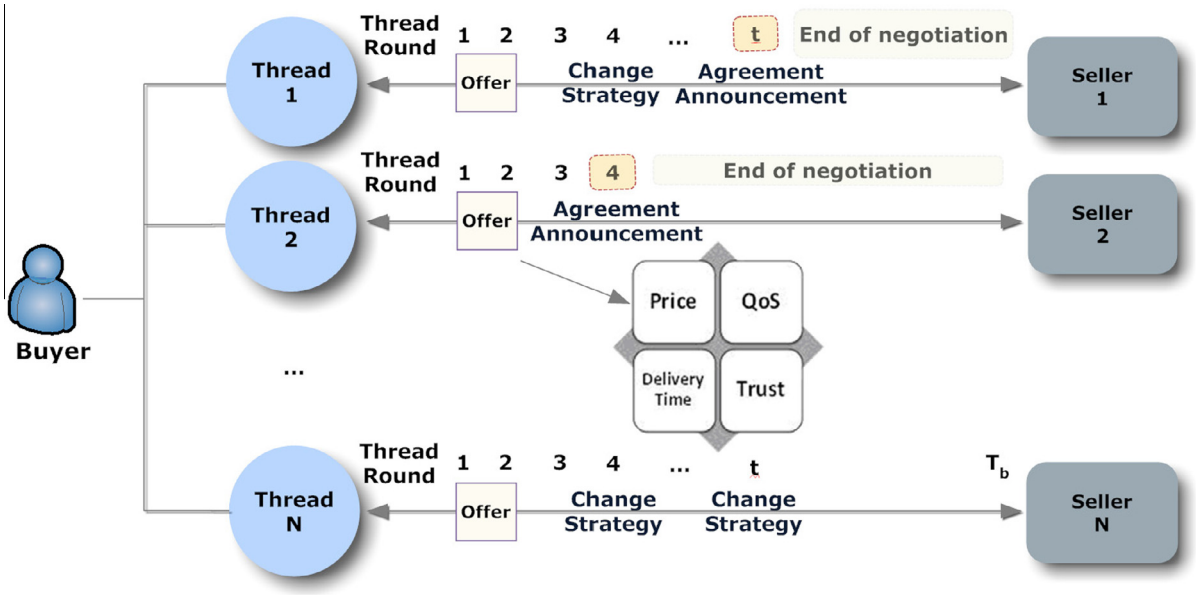

Fig. 1. Concurrent negotiations scenario.

change their strategy to pursue a better agreement than the previous one. In addition, the remaining threads change the weights for the utility calculation to pay more attention on specific issues and, thus, to achieve a better utility in a possible future agreement.

3.2. Entities behavior

We consider that every seller has the same item in her property retrieved with a specific cost that tries to sell it in the highest possible utility. Similarly, the buyer is interested in purchasing the item that is close to her preferences. The item has a number of characteristics (issues) that affect the final utility. These issues are categorized as *proportional* (P) or *inversely proportional* (IP) to the utility. When examining P issues, the higher the issue value is, the higher the utility becomes. The opposite stands for the IP issues. When an issue is characterized as P for the buyer, it is also characterized as IP for the seller and vice versa. The reason is that the buyer and the seller have conflicting and opposite goals. For instance, the item price is characterized as IP for the buyer because, the lower the price is, the higher the buyer's utility becomes. On the other hand, the item price is characterized as P for the seller as a high price leads to high a seller's utility.

The buyer has a specific deadline defined by her owner. The same stands for the seller. Let us denote the buyer deadline with T_b while the seller deadline is depicted by T_s. In each negotiation, the seller starts first and the buyer follows, if the proposed offer is rejected. The seller proposes an offer at odd rounds and the buyer (i.e., her threads) makes a counter offer at even rounds. If an entity is not satisfied by the offer, she has the right to reject it and issue a counter offer. Every offer involves specific values for the examined issues. This approach is defined as the package deal (Rahwan et al., 2002; Torroni and Toni, 2001). If a deadline expires and no agreement is present, the negotiation ends with zero utility for both entities. Entities adopt a specific utility function (U) defined as follows:

$$U = \sum_{i=1}^{m} w_i \cdot v_i \tag{1}$$

where m is the number of issues, w_i and v_i are issues' weights and values, respectively. In addition, both entities have their own strategy for the calculation of offers. Each entity has her own reservation values for every issue. We consider an interval $[min_i, max_i]$ where the ith issue takes its values (Fatima et al., 2005; Oprea, 2002). These values differ in the buyer as well as in the seller side. Both entities generate their offers based on the following equations:

$$O_i = min_i + \varphi(t) \cdot (max_i - min_i) \tag{2}$$

for the buyer and

$$O_i = min_i + (1 - \varphi(t)) \cdot (max_i - min_i) \tag{3}$$

for the seller, respectively. In the above equations, O_i depicts the next offer for issue I, min_i (max_i) is the minimum (maximum) allowable value of issue i and $\varphi(t)$ is a time dependent strategy function. Functions should ensure that $0 \leqslant \varphi(t) \leqslant 1$, $\varphi(0) = k$ and $\varphi(T) = 1$ (T: deadline). Function $\varphi(t)$ is defined as follows (Fatima et al., 2005):

$$\varphi(t) = k + (1 - k)\left(\frac{\min(t, T)}{T}\right)^{1/\psi} \tag{4}$$

An infinite number of functions can be derived for different values of ψ. Finally, for every issue, we calculate the corresponding utility (for both entities) based on the following equations:

$$U(v_i) = \frac{v_i - min_i}{max_i - min_i}, \text{if issue i is P} \tag{5}$$

$$U(v_i) = \frac{\max_i - v_i}{\max_i - \min_i}, \text{if issue i is IP} \tag{6}$$

We propose a model where the buyer threads are automatically organized to change their strategy to reach the best agreement. Our approach is based on the known ABC algorithm. We adopt the ABC algorithm for defining the weights adopted in the utility function. The aim is to have a tradeoff between issues in order to have a better agreement. When a thread concludes an agreement, a message is sent to the rest of the threads. The remaining threads, if necessary (if the utility of the agreement is higher than the utility defined by the current offer), reassess weights w_i aiming at high U. If the weight of an issue, which has higher value than the corresponding issue of the agreement, decreases, its proportion in the utility will decrease too. Through this approach, we emphasize on those issues that have value worse than the agreement realiza-

tion. The reason is that the utility will increase depending on such issues (the utility will be increased if we increase the weight of the specific issue and achieve a better value in the negotiation). This way, the thread tries to force the seller to give better offers than the previous. U will be increased if and only if the seller will improve those values.

4. Threads self adaptation model

4.1. Swarm intelligence theory

The buyer has identified a set of potential sellers and she is involved in negotiations with all of them. The aim is to buy the item accepting an offer that maximizes her utility. For negotiating with all the sellers together, numerous threads are spawned. For all of them there is a common goal: the best possible agreement for the specific set of sellers. Threads could adopt a collective behavior to reach the common goal. By adopting the collective behavior, threads could be beneficial from the negotiation information present in their peers (other threads). Such information could affect their behavior to have the best possible result. These interactions lead to an intelligent behavior fully adapted on the information accumulated by the negotiations underway. Based on this rationale, we adopt SI as the representative theory of collective intelligence. The proposed model is (Bonaneau and Meyer, 2001):

- ***Robust.*** It is not affected by failures. Even if a thread fails to have a successful conclusion, the rest of them can still perform their task.
- ***Flexible.*** The model can react in environment's changes (e.g., change in a seller's strategy).
- ***Self-organized.*** Each thread can easily change its strategy without complex calculations and signaling overhead between threads and the CM.

4.2. The ABC algorithm

As mentioned, the ABC algorithm is adopted for resulting issue weights just after every agreement is announced by a thread. The thread concluding an agreement announces it to the remaining threads and sends a lightweight message containing the agreement information (i.e., values for each issue and the final utility). The remaining threads compare their current negotiation information with the concluded agreement and, if necessary, change the weights of U to pursue a better agreement. The aim is to derive weights that '*pay attention*' in some of the issues to lead to a better agreements (high U) in the upcoming rounds. The rationale is that we try to find weights such as the final U is higher than the utility gained by the latest agreement. With this approach, we emphasize on those issues that have value 'worse' than the values present in the agreement, because the utility will increase when these 'bad' issues will be depicted by high/low (it depends if an issue is P or IP) values. This way, there is a dynamic change in threads' strategy under incomplete knowledge. Let us discuss a simple example depicting our idea. We consider two issues being IP. These issues are the price (p) and the delivery time (d). Both of them should be low to have the buyer tasting a high U (for simplicity, we consider values in the interval [0,1]). Let the negotiation information for thread A be: $p = 0.6$, $d = 0.5$ and let thread B conclude an agreement with $p = 0.7$, $d = 0.5$. In addition, weights for thread A are 0.3 for p and 0.7 for d while thread B adopts 0.4 for p and 0.6 for d, respectively. The utility that the buyer tastes from the agreement in thread B is 0.64. After the conclusion of the agreement, thread A executes our algorithm and changes the adopted weights. If the adapted weights are 0.7 for p and 0.3 for d, thread A will pay more attention on p as d is already lower than the d value realized in the agreement announced by thread B. Hence, if thread A manages to conclude a new agreement with $p = 0.75$ (slightly better than p realization in thread's B agreement), the final utility will be 0.68 (higher than the agreement's utility).

The ABC algorithm resembles the foraging operation of honeybees and their swarming around the hive. The interaction is between three types of bees: *employed*, *onlooker* and *scout*. The ABC algorithm is an iterative process and requires five user parameters: (a) number of food sources (solutions i.e., utility function weight values), (b) number of iterations (MCN), (c) number of cycles before a constant solution (with no improvement) is replaced by a new one, (d) modification rate (MR) that controls the number of parameters to be modified and (e) scout production period (SPP). The number of employed and onlooker bees are set equal to the number of solutions i.e., an employed bee corresponds for every food source. The initial solutions are randomly selected as follows:

$$x_{ij} = LB_j + (UB_j - LB_j)\psi_{ij},\ j = 1, 2, \ldots, n \text{ and } i = 1, 2, \ldots, SN \quad (7)$$

where LB_j and UB_j is the minimum and maximum values of dimension j, ψ_{ij} is a uniformly distributed random number in [0,1] and SN is the colony size. The employed bees are sent to the initial sources, evaluate their fitness functions and return to their hive to inform the bees waiting on the dance area about the amount of nectar of the examined sources. At the next step, the employed bees return to the last known sources and chose a new source in this neighborhood. A uniformly distributed random number, ($0 \leqslant R_j \leqslant 1$), is produced for each parameter x_{ij}. The parameter x_{ij} is then modified if the random number is less than MR.

$$z_{ij} = \begin{cases} x_{ij} + (x_{ij} - x_{kj})\omega_{ij}, \text{if } R_j < MR \\ x_{ij}, \text{ otherwise} \end{cases} \quad (8)$$

where $k \in \{1, 2, \ldots, SN\}$, with $k \neq I$, is a randomly chosen index and ω_{ij} is a uniformly distributed random number in [−1, 1]. If no parameter is changed, one random parameter of the solution is changed as follows:

$$z_{ij} = x_{ij} + (x_{ij} - x_{kj})\omega_{ij}, j = 1, 2, \ldots, n \text{ and } k = 1, 2, \ldots, SN, k \neq i \quad (9)$$

where x_{ij} is the current position of the employed bee, j is a uniformly distributed random integer in the range [1, D] (D is the number of optimization parameters) and ω_{ij} is a uniformly distributed random number in [−1, 1]. It should be noted that the deviation from the current position x_{ij} decreases as the difference between x_{ij} and x_{kj} decreases. After the production of the new position, the ABC makes a selection using Deb's rules (Deb, 2000). Using Deb's rules, the new position is accepted or rejected. According to Deb's method, the following decision scheme is adopted:

- Any feasible solution ($vl_i \leqslant 0$) is preferred to any infeasible solution ($vl_j > 0$);
- Among two feasible solutions ($vl_i \leqslant 0$, $vl_j \leqslant 0$), the one having a better objective function value is preferred ($f_i < f_j$);
- Among two infeasible solutions ($vl_i > 0$, $vl_j > 0$), the one having smaller constraint violation is preferred ($vl_i < vl_j$).

An onlooker bee, then, selects a food source x_i by calculating its probability:

$$p_i = \begin{cases} 0.5 + 0.5 \cdot \left(\frac{ft_i}{\sum_{j=1}^{SN} ft_j} \right), & \text{for feasible solutions} \\ 0.5 \cdot \left(1 - \frac{vl_i}{\sum_{j=1}^{SN} vl_j} \right), & \text{for unfeasible solutions} \end{cases} \quad (10)$$

where vl_i is a penalty and ft_i is the fitness function of source i (the nectar information gathered by the employed bees).

$$ft_i = \begin{cases} \frac{1}{1+f_i}, & \text{if } f_i \geqslant 0 \\ 1 + |f_i|, & \text{if } f_i < 0 \end{cases} \quad (11)$$

where f_i is the cost value of the source i. Infeasible solutions' probabilities are in [0, 0.5] while feasible solutions' probabilities are in the interval (0.5, 1]. Feasible solutions are selected probabilistically proportional to their fitness while infeasible ones are chosen inversely proportional to their violation values. Similar to the employed bees, the onlookers generate a new source using Eq. (8) which is finally selected with a probability related to its nectar amount. If a source cannot be further improved in a predetermined number of cycles, it is abandoned and replaced with a new one produced by scouts through Eq. (7). It should be mentioned that scouts are produced at predetermined periods which are defined by the control parameter known as *scout production period* (SPP). At each SPP cycle, a scout production process is performed if there is an abandoned food source exceeding a pre-defined limit.

4.3. Threads decision process

Threads start from the same strategy related to the generation of offers. The adopted strategies are depicted by Eqs. (2) and (3). If an agreement is concluded, the specific thread sends an agreement message to the rest. We consider that the communication time is negligible. The message conveys information related to the issues under consideration and, thus, the remaining threads can 'see' the utility that the buyer gains. Based on these values and the utility gained by the agreement, the remaining threads can readjust, if needed, their strategies. We consider that threads after the reception of an agreement message check their current utility (i.e., as realized till the specific round of the negotiation) and if the current utility is smaller than the utility gained by the recent agreement, they decide to readjust the weights adopted in Eq. (1). The reason is that by readjusting the weights in the utility function, they readjust the adopted strategy concerning the offers made to the seller as threads aim to gain more utility in the future. The ABC algorithm, as described in Section 4.2, is adopted to result the new weights. Actually, the solutions x_{ij} are realizations of the weights. After that, each thread continues the negotiation, however, following, a new strategy towards the utility maximization.

5. Experimental evaluation

In the previous sections, we analyzed the decision process of the described negotiation scenario. The described model is applied in each thread. Therefore, at every round each thread should accept a seller's offer if the utility of the incoming offer is higher than the utility realized on the thread's next offer (in the case where the seller accepts the offer). In this section, we discuss our experimental results. Our objective is to provide simulation results for very important parameters like average agreements, the buyer's and the seller's utility and the time required to conclude an agreement. These metrics are studied for various thread numbers. We can see how the number of threads affects the agreement price and, thus, the final utility of the buyer. We compare our model with models found in the respective literature. The comparison is related to the final buyer's utility, the agreement ratio (percentage of agreements) and the time required to realize an agreement.

5.1. Performance metrics

The performance metrics adopted in our experimental evaluation are:

- ***The agreement ratio* (*AG*)**: The *AG* metric indicates the number of negotiations that end with an agreement out of a number of negotiations. Let us denote with R the total number of negotiations, with B the number of negotiations where the buyer accepts the seller's offer and with S the number of negotiations where the seller accepts the buyer's offer. The following equation holds true:

$$AG = \frac{B + S}{R} \quad (12)$$

  The higher the *AG* value is, the more agreements are concluded and both entities gain some utility.

- ***Average Buyer Utility* (*ABU*)**: The *ABU* metric indicates the utility that the buyer gains from a negotiation. The utility is calculated when an agreement is present. For the *ABU* metric, the following equation holds true:

$$ABU = \frac{1}{B + S} \sum_{i=1}^{B+S} \max_i(U_b) \quad (13)$$

  The *ABU* is calculated by taking into consideration the maximum utility from the sub-set of threads that concluded an agreement. This is natural as the buyer finally chooses the best agreement among all the threads. We assume that when an agreement is present in a specific thread, the buyer does not announce the conclusion of the negotiation to the corresponding seller, however, she waits for the conclusion of the entire set of threads. We consider that this time does not affect the final conclusion of an agreement.

- ***Average Seller Utility* (*ASU*)**: The *ASU* metric is defined as the utility gained in the seller side. With the ASU metric, we consider the average utility for a number of negotiations. The discussed utility is calculated only in the case of an agreement. The following equation stands true:

$$ASU = \frac{1}{B + S} \sum_{i=1}^{B+S} U_s \quad (14)$$

- ***Average Rounds* (*AR*)**: The *AR* metric is defined as the number of rounds required to reach an agreement out of the full horizon $T = \min(T_b, T_s)$. It is an indication of the time required and the resources respectively to conclude a negotiation. The higher the *AR* is, the more time and resources are required by a negotiation. Actually, we examine the percentage of the required rounds on T. The *AR* metric is defined by:

$$AP = \frac{1}{B + \Sigma} \sum_{i=1}^{B+\Sigma} \frac{Z \cdot t^* + (1 - Z) \cdot t^{\#}}{T} \quad (15)$$

where

$$Z = \begin{cases} 1, \textit{if the buyer accepts the seller's offer at time } t^{*} \\ 0, \textit{if the seller accepts the buyer's offer at time } t^{\#} \end{cases} \quad (16)$$

5.2. Simulation setup

We run a number of simulations for the discussed scenario. The simulations run in an Intel Pentium 3.2 GHz processor with 4 GB Ram running Windows 7. A virtual EM was created, where buyer threads are implemented as Java threads. All threads share the same deadline. We run experiments adopting synthetic data for different values of the buyer valuation (V) about an item. This value affects the item price. We run 100 negotiations for $N_T = 50$ (threads), $I = 4$ (issues – including price) and $V \in \{10, 300\}$. It should be noted that at the beginning of each experiment, we randomly (uniformly) choose intervals $[min_i, max_i]$ for each side (buyer and seller). These values are chosen in [0,100]. Moreover, we randomly choose T_b and T_s in the interval [50,100]. The seller's parameters are also randomly selected in every experiment (e.g., the cost is randomly (uniformly) selected in the interval (Crampton, 1984; Shandholm and Vulkan, 1999). For offers calculation, we are based on Eqs. (2) and (3). It should be noted that with the selected intervals, we try to simulate different scenarios related to the *agreement zone* (Kolomvatsos et al., 2012). The agreement zone is defined as the difference between V and the seller's cost. The agreement zone indicates if there is plenty of room to realize an agreement. For instance, when V is randomly selected in An et al. (2006), Crampton (1984) and the seller's cost is randomly selected in the interval (Crampton, 1984; Shandholm and Vulkan, 1999), we simulate a '*narrow*' agreement zone. In this case, we simulate a scenario where both entities (i.e., buyer threads and the corresponding sellers) should 'consume' their deadlines and accept less profitable offers to, finally, conclude an agreement (even with limited profits for both). In addition, when limited agreement zones are the subject, there is an increased risk of a conflict, especially when the agreement zone is the empty set. When V is high (e.g., $V = 300$), there is plenty of room for an agreement and both entities can pursue very profitable offers. Finally, to depict the performance of our model in real scenarios, we, additionally, execute a set of experiments adopting real data.[2] The adopted dataset provides stock values for S&P 500 containing one record per line where date, open, high, low and close values for each stock are depicted. We adopt *high*, *low* and *close* values for representing the issues involved in the multi-issue negotiations. A subset is adopted for the buyer part while another is adopted for the seller part. We envision that each stock value is the price of product with a current value, i.e., a *high* value and a *low* value. Negotiation begins with the low value and cannot exceed the maximum pre-defined value. At each round, the proposed value of the IA (i.e., the buyer and the seller) is connected with the current value of the stock. The rest issues are calculated as we have already described.

We compare our results with the results derived by another optimization technique that adopts the Simplex method in combination with the PSO algorithm (Panagidi et al., 2013). The simulation setup is as described above for both models. In addition, we present a comparative evaluation of our model with two other algorithms dealing with concurrent negotiations. The *Conan* model is a heuristic strategy for concurrent, single-issue negotiations (Bedour et al., 2014). IAs are autonomous during the negotiation with the absence of a CM. While a thread negotiates with a specific seller, the thread can commit (hold) a preferred offer for a certain amount of time, unless a better offer is received. In such cases, the thread can de-commit and pay a penalty. IAs generate offers by calculating environmental and self (individual) factors. These factors are based on local and global criteria of the negotiation like the number of the committed offers, the number of competitors, the deadline and the individual's negotiation status. The second model use IAs to concurrently negotiate with multiple, unknown opponents in real-time over multi-issues (Williams et al., 2012). The negotiation protocol is orchestrated by a CM. Let us name this model as the *Coordinator* mechanism. Each negotiation thread is responsible for managing the negotiation with a single opponent and uses information learnt during the interaction. In each round, every thread provides the information of the negotiation to the CM. The CM based on probabilistic information about the opponent selects the information from all threads, determines the best response across the entire set of the components and broadcasts this information, i.e., the optimal utility U and the best time t for each negotiation thread i, back to the threads. This information is adopted by each negotiation thread for creating the next offer and updating the termination time of the negotiation.

5.3. Performance results

The proposed algorithm is fast and efficient concerning the time required to conclude an agreement and the performance related to the utility gained by the buyer. The complexity of our algorithm, in the worst case, is $O(T_b \cdot CS)$ where the CS depicts the population size (the number of bees). We observe that the time complexity depends not only of the deadline that the buyer and, thus, threads, but also on the bees number that have the responsibility of performing changes in the weights of the utility function. In the aforementioned complexity, we consider that each bee adopts a single cycle to produce the optimal solution. However, this cannot be efficient as, with a single cycle, the solutions space is not searched exhaustively. Hence, the final complexity of our algorithm is $O(T_b \cdot CS \cdot CL)$ where the CL depicts the cycles that bees adopt to conclude the optimal solution.

In Fig. 2, we can see our results for different V. V, in combination with sellers' cost, affects the agreement zone as already mentioned. The ABC algorithm provides better results related to the adopted metrics (except *ABU*; $V > 150$). The interesting is that the proposed model performs well when $V \to 10$. In such cases, the agreement zone (concerning the price) is limited. The weights adaptation scheme leads to more agreements compared to the PSO as the solution space is optimally searched and new strategies are adopted by threads. The ABC is not trapped in local minima and, thus, it is able to find the optimal solution for the specific information of the negotiation. The increased number of agreements is accompanied by an increased *ABU*, especially, in limited agreement zones. Threads change the adopted weights and, thus, dynamically adapt their strategies to the announced agreements. This has the potential to seek agreements that are better than the previous and, thus, the utility that the buyer gains, is always increasing. Concerning the *ABU*, the proposed model exhibits a stable behavior with $ABU \sim 0.8$. The PSO algorithm achieves better results for high V. However, in such cases the agreement zone is very large and, thus, more possibilities for an agreement are present. Concerning the *AR*, we see that the proposed model reaches an agreement in lower amount of time compared with the PSO. The ABC algorithm achieves 73% (approximately) reduction in the rounds required for an agreement. This is natural as the ABC performs an aggressive weights adaptation to achieve high U compared to past agreements. Through the adaptation process, the focus is paid on different issues that are judged to be lower than the issues realization in recent agreements.

[2] *Historical Data for S&P 500 Stocks, available at* http://pages.swcp.com/stocks/#historical%20data.

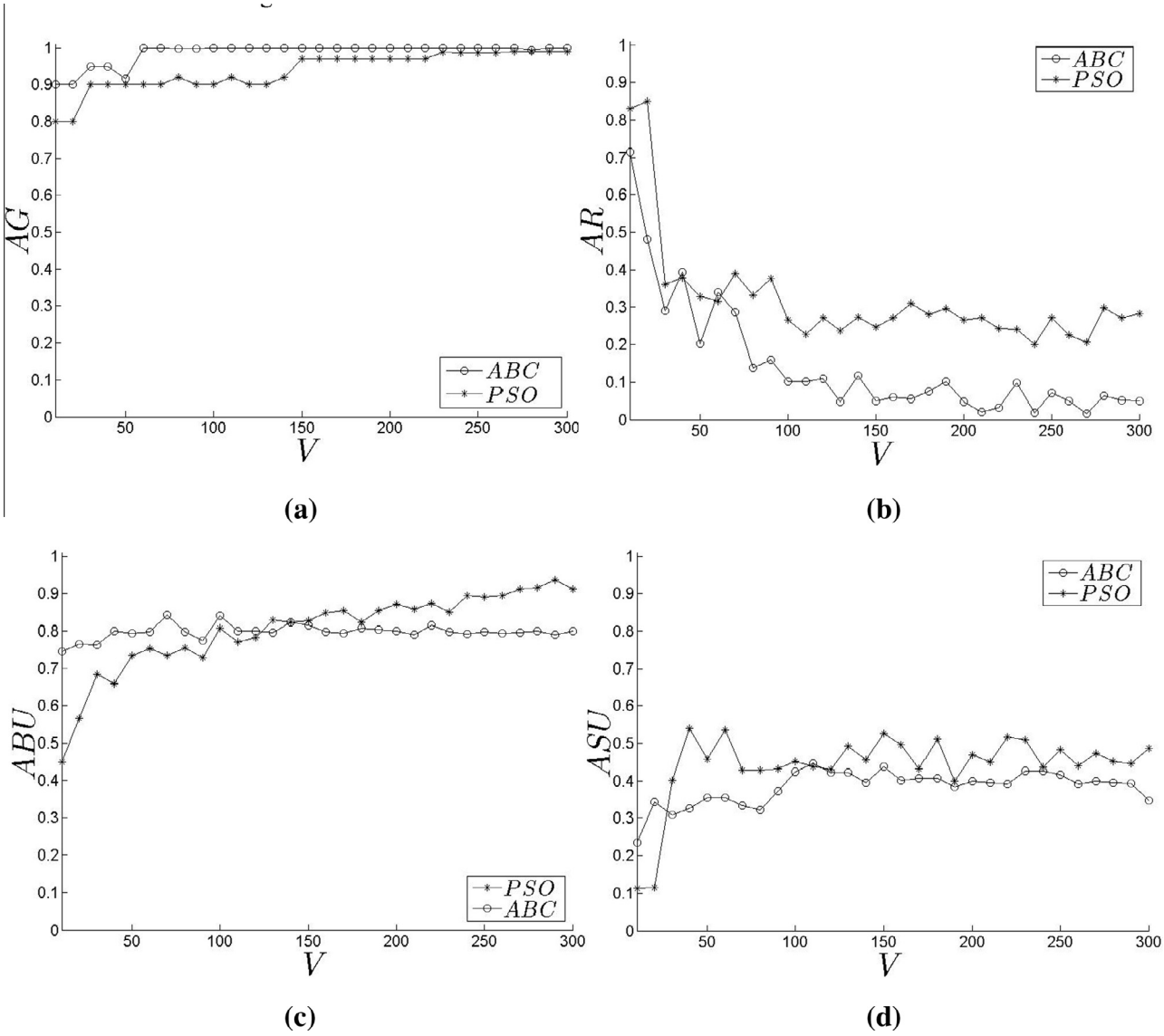

Fig. 2. Results for different V values.

Apart from V, we also focus on the number of threads N_T. We run experiments for different N_T and report on our results. We run 100 simulations keeping $V = 100$, $I = 4$ and $N_T \in \{10, 20, \ldots, 50\}$. Our aim is to identify the effect of N_T in the negotiation outcome. In Fig. 3, we observe similar results as in Fig. 2 (i.e., experiments realized for different V). The ABC approach overcomes the PSO model concerning the *AG*, *ABU* and *AR*. The *AG* and *ABU* increase as N_T increases as well. The ABC model leads to a 31% (approximately) increase in the *ABU* value. This means that the higher the number of threads is, the higher the utility becomes. Through a large number of threads, the buyer has more opportunities to find the best agreement as she faces many sellers. Recall that every new agreement is better than previous agreements realized through the weights adaptation scheme. Hence, the more the number of threads is, the more the opportunities for agreements and increased utility, respectively. When a high number of threads is the case, a new agreement will affect the strategy of many threads and, thus, the buyer has the opportunity to force more sellers to proceed with better offers. The result is that any new agreement, statistically, will involve offers better than previous agreements. We observe this trend in *ABU* results. When $N_T \rightarrow 50$, *ABU* exceeds 0.8 which means that the final agreement can approach the optimal scenario involving issues realization that are very profitable for the buyer.

We, additionally, experiment with real data. We run 100 simulations keeping $V = 100$, $I = 4$ and $N_T \in \{10, 20, \ldots, 50\}$. In Fig. 4, we see our results. The ABC outperforms the PSO and leads to increased agreements and utility for the buyer. In these plots, we also depict the mean (*meanABC*, *meanPSO*) and the deviation of each model. The proposed model exhibits stability in its performance, especially, in the case of *AG* and *AR*. The *AG* is high and the *AR* is low meaning that the proposed model concludes a high number of agreements in limited time. These results confirm the efficiency of the proposed scheme and show us that the aggressive adaptation of weights seems to be very important for the realization of multiple agreements. The interesting is that our model when applied on top of real data leads to lower *ABU* compared to the PSO. In addition, our results on top of real data show that the *ABU* is lower than the utility realized in our experiments involving synthetic data. It should be noted that in our real dataset, the collected values are realized in short intervals and, thus, significant fluctuations in issues realization are limited. The real dataset is depicted by a 'stability' in data fluctuations. Our model seems to be the appropriate solution for a buyer when the underlying data

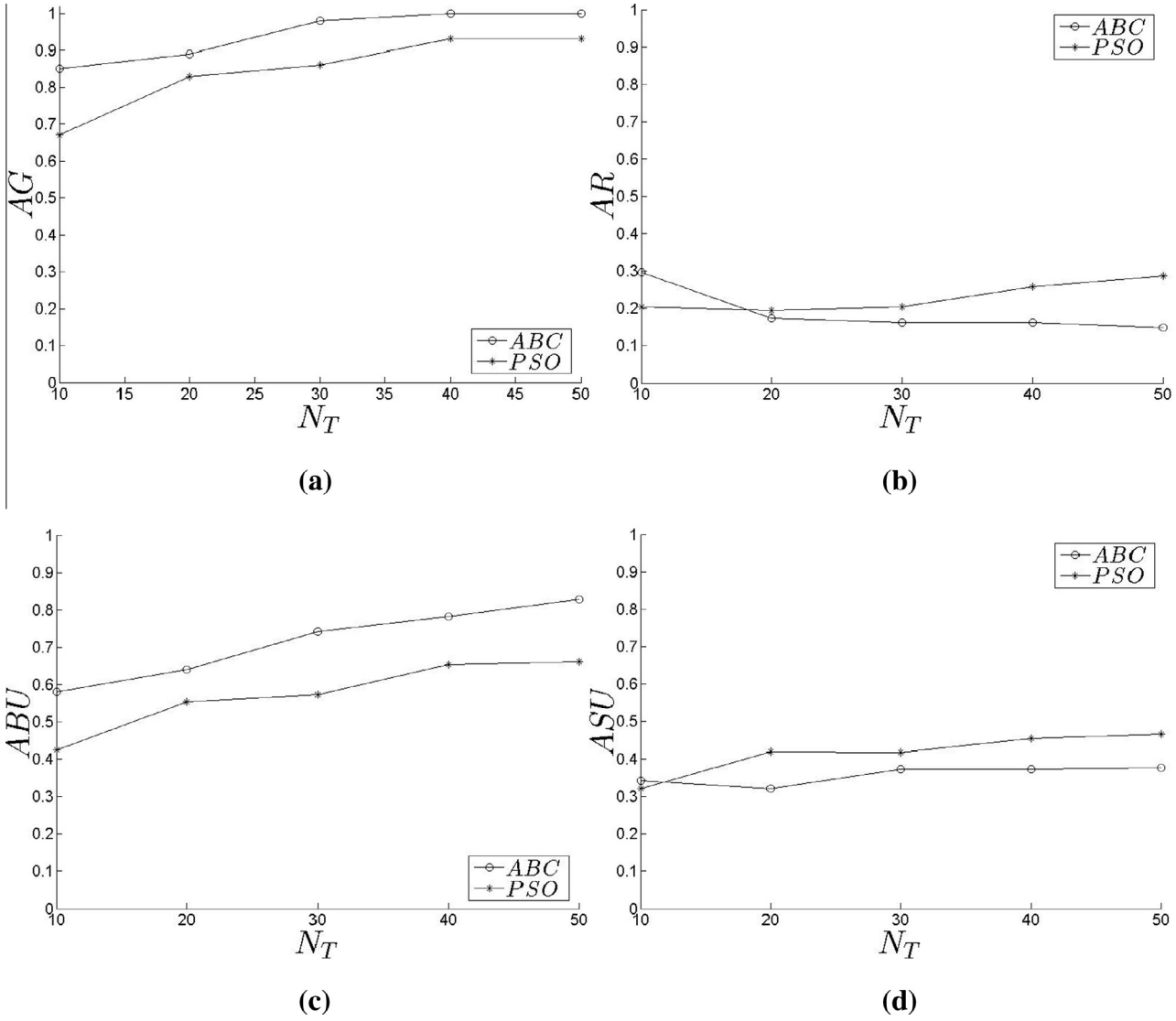

Fig. 3. Results for different N_T values.

are characterized by a uniform distribution where each data value is of equal probability and the environment is very dynamic (data change continually). Hence, the proposed model manages to perform better than the remaining schemes when sellers' offers and strategies are characterized by high fluctuations.

Let us now proceed to a comparison between the proposed model and the PSO model adopting simple statistical metrics. We report on the performance of the discussed models concerning their mean (μ) and standard deviation (σ) while we perform a *t*-test to present the difference in their results. In Table 1, we present our results for the entire set of the adopted metrics. We observe that the proposed models is more stable than the PSO for the *AG*, *ABU* and *ASU* while PSO performs better concerning the *ASU*. The mean value of the ABC related to the *AG* is higher than the PSO while our model also outperforms the PSO concerning the *AR*. Concerning the *ABU*, we observe a similar behavior for both models.

The *t*-test is concluded for both, the synthetic and the real dataset. The critical values for the confidence intervals of 90%, 95% and 99% are 1.310, 1.697 and 2.457, respectively. These values stand for the experiments involving the synthetic dataset and different *V*. Our t-values are 0.211, 0.031, 0.200 and 0.346 for *AG*, *ABU*, *ASU*, and *AR*, respectively. We see that these results do not depict a significant difference in the performance of the examined models that could cause a violation of the critical values. In addition, we perform the *t*-test for the synthetic and the real datasets and for different N_T. The critical values are 1.476, 2.015 and 3.365 for the confidence intervals of 90%, 95% and 99%, respectively. Our results, for the synthetic dataset, are 1.256, 2.032, 0.519 and 1.447 for *AG*, *ABU*, *ASU* and *AR*, respectively. The observed results for the real dataset are 0.997, 0.865, 0.439 and 27.224 for *AG*, *ABU*, *ASU* and *AR*, respectively. We observe a significant difference in the performance of the models concerning the *AG*, *ABU* and *AR* depending on the dataset. Recall that the synthetic dataset is generated by incorporating random (we adopt a uniform distribution) values for the examined issues and, thus, it is characterized by fluctuations on the retrieved values. On the other hand, the real dataset is characterized by stability due to the fact that one cannot observe any high fluctuations in consecutive stock prices. Our model performs better when the dataset is not 'stable'. We observe that critical values are violated for: (i) the *ABU* metric (synthetic dataset) having the *AG* and *AR* *t*-test results close to the critical value (for the same dataset); (ii) the *AR* metric (real dataset). This means that the proposed model outperforms the PSO model concerning the time required to conclude an agreement while having the remaining metrics at high

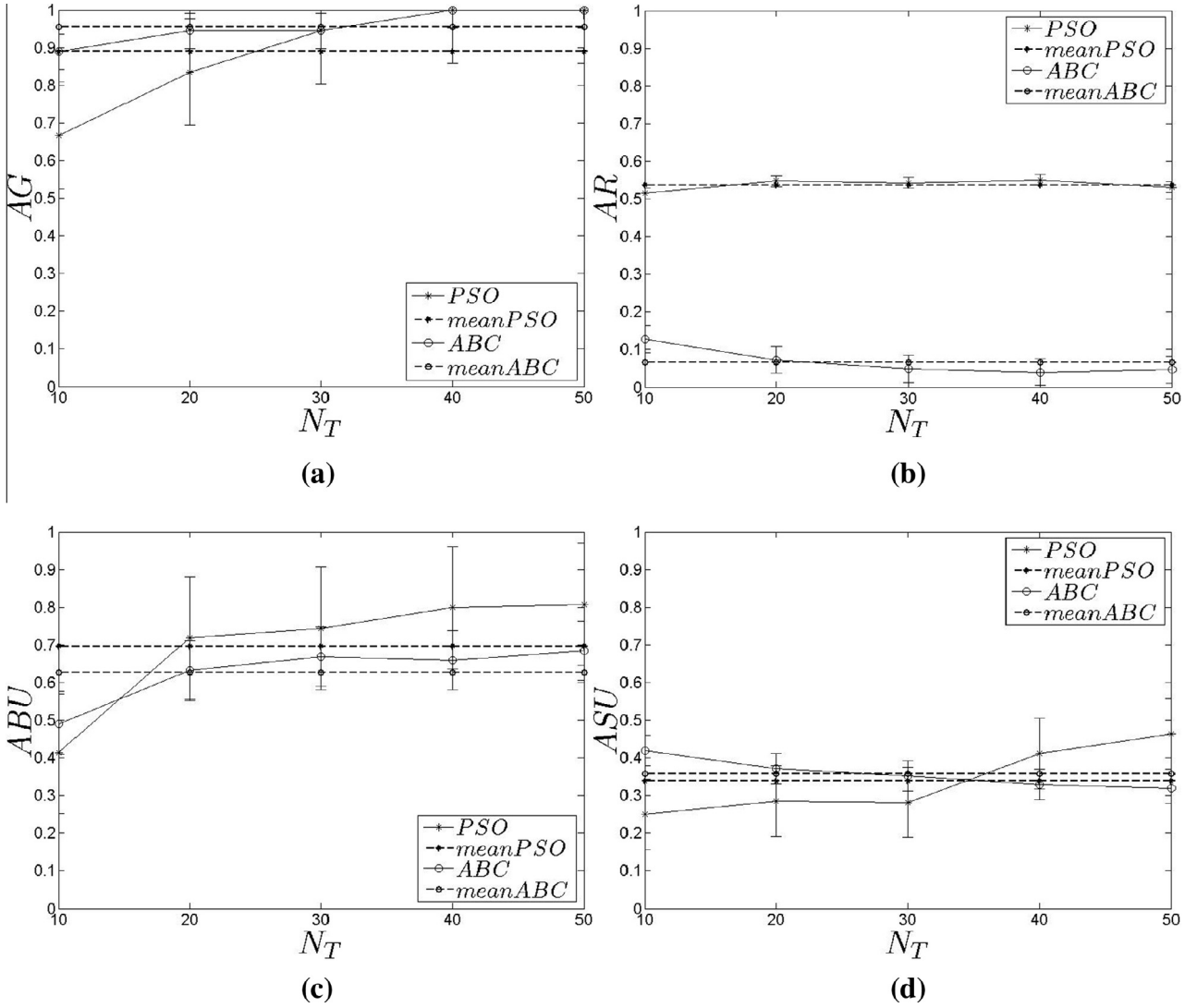

Fig. 4. Results for different N_T values (real data).

Table 1
Mean and standard deviation of performance metrics.

		AG	ABU	ASU	AR
μ	ABC	0.99	0.80	0.38	0.14
	PSO	0.94	0.80	0.44	0.32
σ	ABC	0.03	0.02	0.05	0.16
	PSO	0.05	0.11	0.09	0.15

levels. Such a result is considered as very profitable for the buyer as with the adoption of the ABC algorithm and the weights adaptation scheme, the buyer can efficiently face multiple and diverse sellers.

We also compare our model with two multi-issue negotiation models found in the literature (i.e., the *Conan* and the *Coordinator* models). In Fig. 5, we present our results (realized on top of synthetic data). We observe that for high *V*, the examined models exhibit similar performance with the PSO performing better than the remaining models for the *ABU* and *ASU* metrics. The ABC performs well concerning the AG metric even for low *V*. The same behavior is observed for Conan, however, Conan is unstable for $V > 50$. Conan and Coordinator models perform well concerning the *AR* metric, however, both models exhibit worse performance related to the *AG* and the *ABU* metrics (for the majority of the agreement zones realized by different *V*). In general, the proposed model behaves well and exhibits the best performance for a high set of experimental scenarios while, at the same time, it avoids any disadvantage of the typical centralized systems as already analyzed above. Hence, it can be the appropriate solution when the environment is characterized by dynamic changes in the underlying data and strategies as depicted by the sellers' behavior.

Finally, we report on the communication cost of the proposed scheme. The communication cost depends on the number of the required messages to conclude a set of negotiations. The required messages are of two types: messages exchanged between buyer and sellers threads (X_M) and messages related to the announcement of agreements in the active threads (G_M). We run 100 negotiations for $N_T = 50$, $I = 4$, $V = 200$ and get the average X_M, G_M.

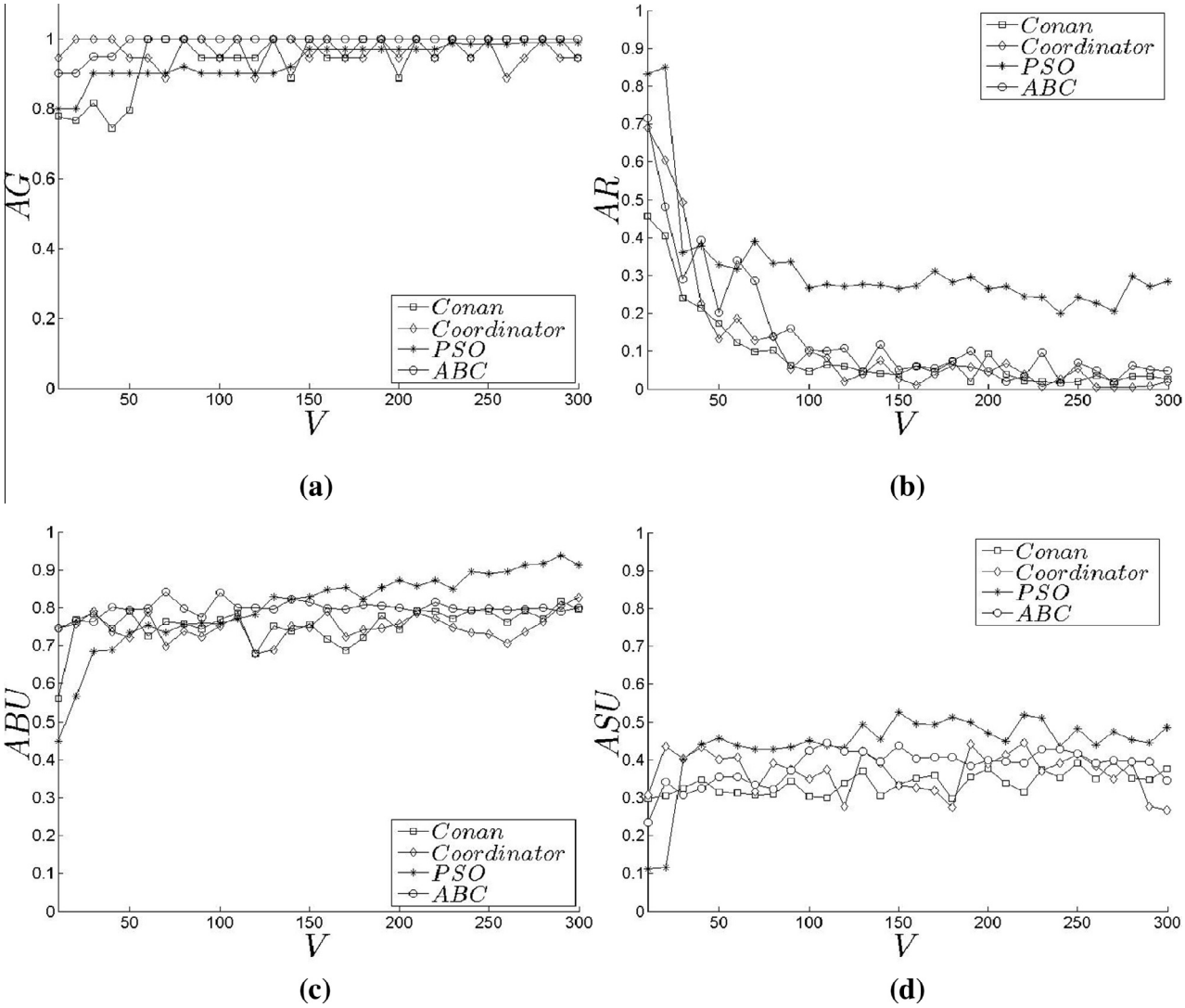

Fig. 5. Comparison results with baseline models.

Concerning the interaction messages, we get $X_M = 20.50$ (in average). This means that the interaction between two threads (buyer–seller) requires in average 21 messages through which values for the examined issues are exchanged before the negotiation concludes (no matter the result). Concerning the agreement messages, we get $G_M = 2.40$. This means that, in average, the agreement messages are sent to two active threads that continue the discussed interactions. It should be noted that the aforementioned values concern results for messages per thread. In general, in every negotiation, the exchanged messages (in average) are 463.92 and 75.58 for X_M and G_M, respectively. We observe that the communication cost, for a high number of threads ($N_T \rightarrow 50$), mainly depends on the messages required by the interaction between buyer – seller and not on the messages exchanged to transfer the agreements information to the active threads. In any case, we could reduce the number of the 'agreement' messages if we consider the announcement of agreements only to a subset of threads according to pre-defined criteria (some nodes could be selected to continue negotiations while others will be stopped). This way, we can reduce the cost for messaging and increase the performance of the algorithm as threads should devote limited resources for communication. However, this approach is left for future work.

6. Conclusions

In this paper, we focus on concurrent negotiations and propose a model for defining weight values in the calculation process of the utility function. We focus on the buyer side and deal with a scenario where the buyer adopts a number of threads for negotiating with a group of sellers. We describe our method for optimizing utility values when an agreement is concluded in a negotiation. The corresponding thread announces the agreement and the rest of them, through the proposed model, try to find the weights that maximize the utility in the upcoming rounds.

Based on the knowledge gained from recent research activities, this paper tries to make one step towards the application of automated negotiations to real world problems. In this context, a dynamic approach was followed where each thread acts independently, being adapted to each seller strategy. To adjust each thread's strategy and reach the optimal deal, the *Artificial Bee Colony* (ABC) algorithm is used, exploiting its strength to better 'escape' from local minima (maxima) independently of the initial values of the examined variables. Moreover, the use of a coordinator specifying the strategy for each thread is avoided to minimize the number of the required resources and the exchanged messages. The coordinator consists of a conceptual single point of failure. If

the coordinator fails to provide the necessary instructions to threads, threads cannot conclude the negotiations. Complex calculations in the coordinator or messaging malfunction cause unnecessary delays. In case of a centralized approach, many messages are required for the communication between the coordinator and threads. Assuming N threads and a deadline of T_b, this approach shows a time complexity of $O(N \cdot T_b)$.

A large number of experiments show that the proposed solution achieves better performance than other research efforts in the field. The buyer utility remains at high levels while the number of agreements reaches 100%. Future work involves the definition of relevant function for weights adaptation in the seller side. Moreover, concerning ABC, a combination of the proposed technique with other intelligent models, like fuzzy logic based mechanisms, could increase the efficiency of the proposed framework.

References

An, B., Sim, K., Tang, L., Li, S.Q., Cheng, D.J., 2006. Continuous-time negotiation mechanism for software agents. IEEE Trans. Syst. Man Cybern. Part B 36, 1261–1272.

Baarslag, T., Fujita, K., Gerding, E., Hidriks, K., Jennings, N., Jonker, C., Kraus, S., Lin, R., Robu, V., Williams, C., 2011. Evaluating practical negotiating agents: results and analysis of the 2007 international competition. Artif. Intell. 198, 73–103.

Bedour, A., Ozgur, K., Kostas, S., 2014. CONAN: a heuristic strategy for concurrent negotiating agents. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems (AAMAS '14). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1585–1586.

Bonaneau, E., Meyer, C., 2001. Swarm intelligence: a whole new way to think about business. Harv. Bus. Rev., 107–114

Bui, H., Venkatesh, S., and Kieronska, D., 1995. An Architecture for Negotiating Agents that Learn, Technical Report. Department of Computer Science Curtin University of Technology, Perth, Australia.

Chatterjee, K., Samuelson, L., 1988. Bargaining under two-sided incomplete information: the unrestricted offers case. Oper. Res. 36 (4), 605–638.

Chen, Y.M., Huang, P.N., 2009. Agent-based bilateral multi-issue negotiation scheme for E-market transactions. Appl. Soft Comput. 9, 1057–1067. Elsevier.

Chen, S., Weiss, G., 2015. An approach to complex agent-based negotiations via effectively modeling unknown opponents. Expert Syst. Appl. 42, 2287–2304.

Cheng, D., Chan, C., Lin, C., 2005. Buyer-supplier negotiation by fuzzy logic based agents. In: Proceedings of the 3rd Int. Conference on Information Technology and Applications. pp. 137–142.

Crampton, P.C., 1984. Bargaining with incomplete information: an infinite-horizon model with two-sided uncertainty. Rev. Econ. Stud., 579–593

Da-Jun, C., Liang-Xian, X., 2002. A negotiation model of incomplete information under time constraints. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, Bologna, Italy. pp. 128–134.

Deb, K., 2000. An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. 186 (2–4), 311–338. Elsevier, Netherlands.

Engelbrecht, A.P., 2007. Computational Intelligence. An Introduction. John Wiley and Sons Ltd.

Faratin, P., Sierra, C., Jennings, N.R., 1998. Negotiation decision function for autonomous agents. Int. J. Rob. Auton. Syst. 24, 159–182.

Fatima, S.S., Wooldridge, M., Jennings, N., 2005. Bargaining with incomplete information. Ann. Math. Artif. Intell. 44 (3), 207–232.

Fudenberg, D., Levine, D.K., Tirole, J., 1987. Incomplete information bargaining with outside opportunities. Q. J. Econ. 102 (1), 37–50. MIT Press.

Gerding, E., van Bragt, D., 2003. Multi-issue negotiation processes by evolutionary simulation, validation and social extensions. Comput. Econ. 22 (1), 39–63.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA.

Haberland, V., Miles, S., Luck, M., 2015. Adjustable Fuzzy Inference for Adaptive Grid Resource Negotiation. Book chapter in Next Frontier in Agent-based Complex Automated Negotiation, vol. 596 of the series Studies in Computational Intelligence. pp. 37–57.

Haberland, V., Miles, S., Luck, M., 2015b. Negotiation strategy for continuous long-term tasks in a grid environment. In: Autonomous Agents and Multi-Agent Systems, pp. 1–21.

Haji, M.H., Gourlay, I., Djemame, K., Dew, P.M., 2005. A SNAP-based community resource broker using a three-phase commit protocol: a performance study. Comput. J. 48 (3), 333–346.

Jazayeriy, H., Azmi-Murad, M., Sulaiman, N., Uzidir, I., 2011. Pareto-optimal algorithm in bilateral automated negotiation. Int. J. Digital Content Technol. Appl. 5 (3), 1–11.

Jonker, C., van der Meij, L., Robu, V., Treur, J., 2004. Demonstration of a Software System for Automated Multi-Attribute Negotiation. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, New York City, USA.

Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization. Erciyes University, Engineering Faculty Computer Engineering Department Kayseri, Türkiye. Technical Report – TR06.

Karaboga, D., Akay, B., 2009. A comparative study of Artificial Bee Colony algorithm. Appl. Math. Comput. 214, 108–132.

Karaboga, D., Ozturk, C., 2011. A novel clustering approach: artificial bee colony (ABC) algorithm. Appl. Soft Comput. 11, 652–657.

Kolomvatsos, K., Hadjiefthymiades, S., 2014. On the use of particle swarm optimization and kernel density estimator in concurrent negotiations. Inf. Sci. (INS) 262, 99–116. Elsevier.

Kolomvatsos, K., Anagnostopoulos, C., Hadjiefthymiades, S., 2008. On the Use of Fuzzy Logic in a Seller Bargaining Game. In: Proceedings of the 32nd IEEE Computer Society International Conference on Computers, Software and Applications (COMPSAC). pp. 184–191.

Kolomvatsos, K., Hadjiefthymiades, S., 2008. Implicit Deadline Calculation for Seller Agent Bargaining in Information Marketplaces. In: Proceedings of the 2nd International Conference on Complex, Intelligent and Software Intensive Systems (CISIS 2008). Barcelona, Spain. pp. 184–190.

Kolomvatsos, K., Hadjiefthymiades, S., 2012. Buyer behaviour adaptation based on a fuzzy logic controller and prediction techniques. Fuzzy Sets Syst. (FSS) 189 (1), 30–52. Elsevier.

Kolomvatsos, K., Trivizakis, D., Hadjiefthymiades, S., 2015. An adaptive fuzzy logic system for automated negotiations. Fuzzy Sets Syst. 269, 135–152.

Lau, R.Y.K., 2005. Towards Genetically Optimized Multi-Agent Multi-Issue Negotiations. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS '05).

Leu, S.S., Sona, P.H., Nhung, P.T.H., 2015. Development of recursive decision making model in bilateral construction procurement negotiation. Autom. Constr. 53, 131–140.

Luo, X., Jennings, N., Shadbolt, N., Leung, H., Lee, J., 2003. A fuzzy constraint based model for bilateral, multi-issue negotiations in semicompetitive environments. Artif. Intell. 148, 53–102.

Narayanan, V., Jennings, N. R., 2005. An adaptive bilateral negotiation model for e-commerce settings. In: Proceedings of the 7th International IEEE Conference on E-Commerce Technology, Munich, Germany. pp. 34–39.

Netto, M.A.S., 2011. CANPRO: A Conflict-Aware Protocol for Negotiation of Cloud Resources and Services. In: Kappel, G., Maamar, Z., Motahari-Nezhad H.R., (Eds.), ICSOC 2011, LNCS 7084. pp. 541–548.

Nguyen T.D., Jennings N.R., 2003. Concurrent bi-lateral negotiation in agent systems. In: Proceedings of the 4th DEXA Workshop on E-Negotiations, Prague, Czech Republic. pp. 844–849.

Nguyen, T.D., Jennings, N.R., 2003. A Heuristic Model for Concurrent Bilateral Negotiations in Incomplete Information Settings. In: Proceedings of the International Joint Conference in Artificial Intelligence. pp. 1467–1469.

Nguyen, T.D., Jennings, N., 2004. Coordinating multiple concurrent negotiations. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems. pp. 1064–1071

Oliver, J.R., 1997. A machine learning approach to automated negotiation and prospects for electronic commerce. J. Manage. Inf. Syst. 13 (3), 83–112.

Oprea, M., 2002. An adaptive negotiation model for agent based electronic commerce. Studies Inf. Control 11 (3), 271–279.

Pan, Q.-K., Tasgetiren, F.M., Suganthan, P.N., Chua, T.J., 2011. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. Inf. Sci. 181, 2455–2468. Elsevier.

Panagidi, K., Kolomvatsos, K., Hadjiefthymiades, S., 2013. On the Use of PSO with Weights Adaptation in Concurrent Multi-Issue Negotiations. In: Proceedings of the 10th International Symposium on Distributed Computing and Artificial Intelligence (DCAI '13), Salamanca, Spain.

Panagidi, K., Kolomvatsos, K., Hadjiefthymiades, S., 2014. An intelligent scheme for concurrent multi-issue negotiations. Int. J. Artif. Intell. (IJAI) 12 (1), 129–149.

Raeesy, Z., Brzostwoski, J., Kowalczyk, R., 2007. Towards a fuzzy-based model for human-like multi-agent negotiation. In: Proceedings of the IEEE/WIC/ACM Conference on Intelligent Agent Technology. pp. 515–519.

Rahwan, I., Kowalczyk, R., Pham, H., 2002. Intelligent agents for automated one-to-many e-commerce negotiation. In: Proceedings of the IEEE International Conference on Privacy, Security and Data Mining. pp. 197–204

Raiffa, H., 1982. The Art and Science of Negotiation. Harvard University Press, Cambridge.

Rajavela, R., Thangarathanam, M., 2016. Adaptive probabilistic behavioural learning system for the effective behavioural decision in cloud trading negotiation market. Future Gener. Comput. Syst. 58, 29–41.

Robu, V., Somefun, D.J.A., La Poutré, J.A., 2005. Modeling Complex Multi-Issue Negotiations using Utility Graphs. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, New York, USA. pp. 280–287.

Shandholm, T., Vulkan, N., 1999. Bargaining with deadlines, 7th Conference on Innovative Applications of Artificial Intelligence, pp. 44–51.

Shyur, H.-J., Shih, H.-S., 2015. Designing a multi-issues negotiation support system based on prospect theory. Inf. Sci. 322, 161–173.

Sim, K.M., 2013. Complex and concurrent negotiations for multiple interrelated e-Markets. IEEE Trans. Syst. Man Cybern. Part B Cybern. 43 (1), 230–245.

Stahl, I., 1972. Bargaining Theory. Stockholm School of Economics, Stockholm, Sweden.

Sun, T., Zhu, Q., Li, S., Zhou, M., 2007. Open, Dynamic and Continuous One-to-Many Negotiation System. In: Proceedings of the 2nd International Conference on Bio-Inspired Computing. pp. 87–93.

Szeto, W.Y., Wu, Y., Ho, S.C., 2011. An artificial bee colony algorithm for the capacitated vehicle routing problem. Eur. J. Oper. Res. 215, 126–135.

Torroni, P., Toni, F., 2001. Extending a logic based one-to-one negotiation framework to one-to-many negotiation. Proceedings of the 2nd International Workshop on Engineering Societies in the Agents World II, pp. 105–118.

Türkay, D., Koray, A., 2012. Modified even-swaps: a novel, clear, rational and an easy-to-use mechanism for multi-issue negotiation. Comput. Ind. Eng. 63 (4), 1013–1029.

Venugopal, S., Chu, X., Buyya, R., 2008. A Negotiation Mechanism for Advance Resource Reservations Using the Alternate Offers Protocol. Quality of Service, 2008. IWQoS 2008. 16th International Workshop on, Enschede. pp. 40–49.

Williams, C., Robu, V., Gerding, E., Jennings, N., 2012. Negotiating Concurrently with Unknown Opponents in Complex, Real-Time Domains. In: Proceedings of the 20th European Conference on Artificial Intelligence, Montpellier, France. pp. 834–839.

Wu, M., Weerdt, M., Poutre, H., 2009. Efficient Methods for Multi Agent Multi-Issue Negotiation: Allocating Resources. In: Proceedings of the 12th International Conference on Principles of Practice in Multiagent Systems. pp. 97–112.

Zeng, D., Sycara, K., 1998. Bayesian learning in negotiation. Int. J. Hum. Comput. Studies 48 (1), 125–141.

Zuo, B.-H., Sun, Y., 2009. Fuzzy Logic to Support Bilateral Agent Negotiation in E-Commerce. In: Proceedings of the 2009 International Conference on Artificial Intelligence and Computational Intelligence, Shanghai, China. pp. 179–183.