Knowl Inf Syst (2018) 56:463–501
https://doi.org/10.1007/s10115-017-1125-2

REGULAR PAPER

# Concurrent bilateral negotiation for open e-markets: the CONAN strategy

Bedour Alrayes[1] · Özgür Kafalı[2] · Kostas Stathis[3]

Received: 16 September 2016 / Revised: 13 July 2017 / Accepted: 10 October 2017 /
Published online: 28 October 2017

**Abstract** We develop a novel strategy that supports software agents to make decisions on how to negotiate for a resource in open and dynamic e-markets. Although existing negotiation strategies offer a number of sophisticated features, including modelling an opponent and negotiating with many opponents simultaneously, they abstract away from the dynamicity of the market and the model that the agent holds for itself in terms of ongoing negotiations, thus ignoring information that increases an agent's utility. Our proposed strategy COncurrent Negotiating AgeNts (CONAN) considers a weighted combination of modelling the market environment and the progress of concurrent negotiations in which the agent partakes. We conduct extensive experiments to evaluate the strategy's performance in various settings where different opponents from the literature provide a competitive market. Our experiments provide statistically significant results showing how CONAN outperforms the state-of-the-art in terms of the utility gained during negotiations.

**Keywords** Automated negotiation · Concurrent negotiation strategy · Electronic markets · Multi-agent systems · Empirical evaluation

✉ Bedour Alrayes
balrayes@ksu.edu.sa

Özgür Kafalı
rkafali@ncsu.edu

Kostas Stathis
kostas.stathis@rhul.ac.uk

1 College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

2 Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206, USA

3 Department of Computer Science, Royal Holloway, University of London, Egham TW20 0EX, UK

## 1 Introduction

Electronic marketplaces (e-markets) such as E-Bay and Amazon are inter-organisational information systems that enable participating buyers and sellers to exchange information about prices and product offerings online, and typically support all steps of the entire order fulfilment process [9]. One problem with existing e-markets is that their participants must monitor continuously how their activities progress. As the duration of some of these activities can be long, e-market participants may find it tedious and time-consuming to monitor multiple, on-going activities [32]. Consider the following scenario:

> Bob needs a second-hand laptop and joins an e-market to purchase it. Bob's preferred price range is between [700–900] and has a deadline to acquire it within five days. He enters the e-market on a Monday and after a quick search, he finds his preferred laptop being sold in an on-line auction. But as on-line auctions are time consuming (indeed the auction's expiration deadline is in a month's time), he also searches for individual sellers who allow items of interest to be negotiated on-line. He finds two possible sellers: Alice and Tom. As shown in Fig. 1, Bob starts a negotiation with both sellers at the same time, by making offers to each one of them separately for the laptops they offer. Alice, sends a counter-offer first, followed by Tom. After evaluating both counter-offers, Bob is inclined to buy the laptop from Tom. But, a new seller, John, joins the e-market on Tuesday. Bob's choices are now extended as the number of alternatives increases because the market has dynamically changed. Should he (a) buy from Tom? (b) negotiate with John risking to lose Tom's offer? or (c) continue to negotiate with all three, in which case what offer should he make to them?

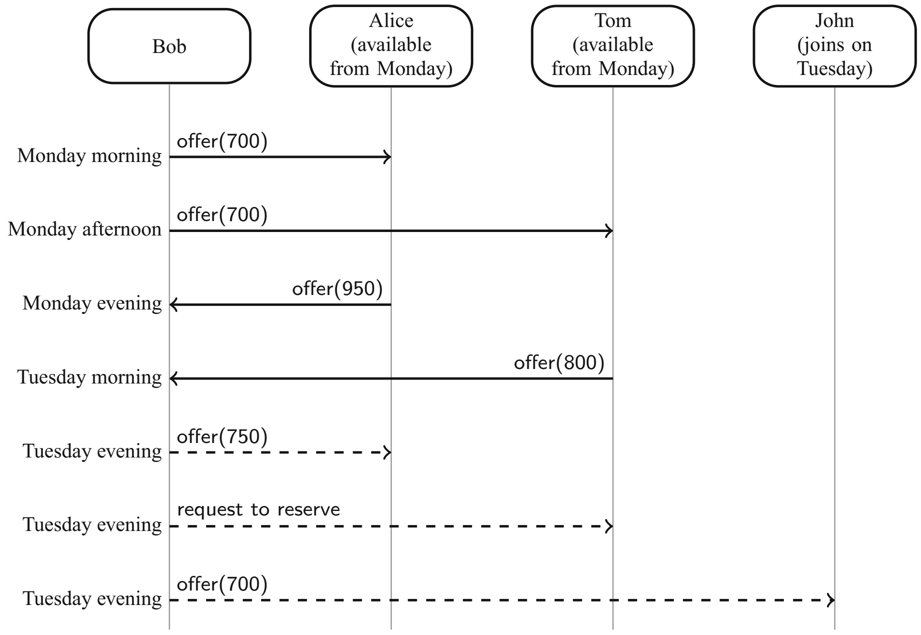


**Fig. 1** Buyer Bob negotiates concurrently with sellers Alice, Tom and John

To answer the questions raised by our scenario, Bob needs a high-level plan, or *strategy*, about which actions to take. This should consider his preferences, the nature of the e-market and the lack of information about the strategies of the sellers, their deadlines or information about possible competitors (other buyers who wish to buy the same laptop and may be negotiating with Bob's sellers at the same time as he is). For the dynamic part of the e-market environment Bob's strategy should also consider if current competitors leave the e-market because their deadlines have been reached or if new competitors have entered the e-market, while Bob is still negotiating. For the incomplete information part too Bob's strategy should consider different issues like how much longer will Tom remain in the market and if there are competitors interested in Tom's laptop, to have an idea of how much to propose to him in his next offer.

Motivated by the above scenario, we study the problem of what kind of strategy should a software agent use in order to represent a buyer's needs in an e-market. We employ a multi-agent systems approach [61] to deal with such dynamic and complex e-markets, whereby agents negotiate bilaterally with others on behalf of their users. The idea here is that agents can satisfy their users' preferences by isolating emotions and feelings [22] as well as save users time. We are interested in many-to-many negotiations that occur in pairs (*bilateral*) and at the same time (*concurrent*). Supporting such negotiations is important because they can enable practically automated buying and selling processes online, e.g. on E-Bay. As agents are inherently multi-threaded, they can negotiate concurrently. The immediate advantage is to eliminate one of the shortcomings of human negotiations, slowness, especially in applications where the task of negotiation can be delegated to a software agent to save time. Another advantage is to evaluate offers from multiple sellers, which increases the chances of reaching an agreement, thus maximising the agent's *utility* [6,47,63]. Utility can be seen as a function that is designed by the user to measure the performance of the agent during negotiation [36]. Moreover, concurrent negotiation provides the agent with negotiating power, which means the agent can negotiate with many sellers at the same time, to acquire a stronger negotiation stance relative to another seller [50,58].

Negotiation has gained attention in numerous multi-agent system applications [37], not only in e-commerce [23,38,52], but also in other areas such as service-oriented systems [13, 45], formation of virtual organisations [43], supply chains [16,27], service grids [60] and cloud computing [15]. In this context, many different aspects of negotiation strategies have been studied, ranging from the introduction and evaluation of opponent models [19,28,39, 47,63,64] to encounter-based learning approaches [40,59]. Negotiation strategies based on game theoretic approaches are often used for practical applications; however, they have been critiqued in situations where one cannot have complete information about the negotiating environment and the negotiation opponents [31,50]. Like [6], we are not assuming that agents have complete information about the factors considered in our framework, which makes agents' reasoning even more difficult to compute optimal negotiation strategies. Therefore, instead of formally modelling the negotiation process, we have followed a heuristic approach that is experimentally tested and evaluated.

In addition, existing strategies abstract away from the e-market environment and its dynamic nature and, as a result, how the environment affects the progress of ongoing negotiations, which in turn lead agents to obtain lower utility. In addition, existing strategies do not always have explicit measures of how well the agent itself is progressing during a negotiation and, at times, they make strong assumptions about the domain, e.g. that deadlines of all participants are public, thus constraining their applicability in a variety of practical negotiation settings. Moreover, the concurrent negotiation literature does not tackle the problem of different agreement zones during negotiation. In the literature, the intersection between the price ranges (known as agreement zones) is always implicitly assumed to be 100% of initial and reservation prices of buyer, an assumption that is unrealistic in practical applications. The reason for exploring different agreement zones is to show how negotiation can move the positions of the agents closer to the areas of the zones where there is agreement and thus improve their utility.

Another major issue related to building practical negotiation strategies for e-markets is how to evaluate the performance of an agent's strategy. While some researchers test their agents theoretically [21], most agent developers use simulation platforms [18,29,39,48,63], especially for evaluating heuristic-based strategies. However, some of these platforms limit the types of strategies that can be tested and as a result there is scope for a standardised simulation environment to provide fair and objective comparisons between negotiating agents.

Given the limitations of existing work discussed above, our hypothesis is that: if an agent models *the e-market environment* and *the progress of itself in it* as explicit parameters, then the agent can capitalise on the fluctuations of these parameters to make offers that improve the agent's utility. To test this hypothesis we propose the CONAN strategy, a heuristic-based strategy that supports a buyer negotiate in open e-market environments, where agents not only enter and leave at their own will, but also have diverse strategies that are unknown to their negotiation opponents. Our metrics to evaluate our hypothesis is based on a novel combination of *environment* and *self*-related factors, which to the best of our knowledge has not been attempted in existing work. Moreover, we keep the deadline of negotiating agents private, so that agents do not know when others will exit from the negotiation. Furthermore, we experiment with a variety of realistic settings by using an agent platform that supports changing the parameters of a simulation to evaluate the performance of CONAN and other state-of-the-art strategies. Our contribution can thus be summarised as follows.

- We formulate the CONAN strategy, and we show how it can be specified declaratively as part of an adaptive agent model for concurrent bilateral negotiations. CONAN makes concurrent bilateral negotiation suitable in applications where existing work imposes too strong assumptions to be realistic.
- We build an open and complex negotiation environment to test the strategy. We then design a series of experiments simulating negotiation scenarios between CONAN and opponent strategies from the literature, and we evaluate extensively their performance.
- We demonstrate that CONAN significantly outperforms the state-of-the-art [63] in terms of average utility gained from negotiations.

The rest of the paper is structured as follows. Section 3 formalises our concurrent negotiation environment. Section 4 describes our negotiating agent, CONAN. Section 5 explains our experimental setting and evaluates the experimental results. Section 2 reviews the relevant literature. Section 6 concludes with potential future directions for our work.

## 2 Related work

We review here related work according to the number of agents negotiating at a specific time, giving rise to multiple concurrent bilateral negotiation models. Our review will focus on addressing the following questions: (a) How an agent can behave in a dynamic environment where there are both external changes (i.e. changes that are out of the agent's control, namely changes in the environment) and internal changes (i.e. changes in the agent's intentions and goals, namely changes in the agent's mental state); (b) when and how an agent can change its strategy during negotiation; and (c) what information is available for an agent during the negotiation to support the decision-making process. In addition, our review will also include two of the concurrent alternating offers protocols that are similar to ours.

## 2.1 Multiple concurrent bilateral negotiation models

In this context Nguyen and Jennings [46,47] developed and evaluated a heuristic concurrent bilateral negotiation model for e-commerce purchasing. The model consists of a *coordinator* and a number of *negotiation threads*. The coordinator is responsible for deciding and changing the negotiation strategy for each thread and updating the lowest utility value of an offer that is considered acceptable by the agent. The coordinator also employs the negotiation strategy according to (a) the *probability distribution of seller*, determining whether the seller is one of conceder or non-conceder; (b) the *percentage of success matrix*, describing the chance of reaching an agreement when the buyer agent applies a specific strategy to negotiate with a particular type of seller; and (c) the *payoff matrix*, measuring the average utility value of an agreement reached in similar situations. The success and payoff matrices are initially set to a common value and updated after the negotiation finishes.

While Nguyen and Jennings [46,47] model is very relevant to our study, it is not directly applicable to our negotiation setting. This is due to their strict assumption about the knowledge of the opponent types, negotiation success and payoff matrices before the negotiation. Moreover, their model is based on a Bayesian learning approach requiring prior knowledge about the environment as well as the opponents. The CONAN strategy presented later in this paper is more open than that of Nguyen and Jennings as it does not deal only with two types of sellers, agents can come and go during negotiation, and it does not require prior information about the opponents and the environment in the form of success and payoff matrices.

An et al. [7] proposed a strategy for deciding what proposal to make and when to make it, in multiple concurrent negotiations. The limitation of this strategy is that compared with our strategy, the negotiation situations of different negotiation threads are calculated according to the minimum offer received from all sellers, while in our case, we analyse each negotiation thread based on how the negotiation with the opponent progresses. Also, in the An et al. strategy, there are no rules to decide when to accept an offer, but in our strategy there are. Moreover, their strategy does not employ a concurrent protocol, as we do. Furthermore, An et al. did not benchmark their strategy against other state-of-the-art strategies, while in our strategy we do.

Li et al. [39] presented a strategy for bilateral negotiations about one issue (i.e. price). Their strategy is composed of sub-strategies about single-threaded negotiations where one buyer and one seller are assumed; synchronised multi-threaded negotiations where there are multiple threads in the buyer account for negotiating with different sellers; and dynamic multi-threaded negotiations where the uncertainty of new sellers joining in the future is factored in to the buyer's strategy. The sub-strategy that is most related to our setting is that of dynamic multi-threaded negotiations in the presence of other opponents. One limitation of the strategy is that it adopts only the basic time-dependent negotiation strategy, viz., the concession rate of the offer only depends on the negotiation time. Furthermore, the work does not provide implementation details on how the authors set up their empirical evaluation. In addition, the strategy assumes the availability of prior information about opponents (e.g. the estimated reservation price of an opponent), which is again not appropriate for practical negotiation settings like ours.

Kolomvatsos et al. [33] adopted particle swarm optimisation (PSO) and kernel density estimation (KDE) for each thread in the concurrent negotiation to find the proposed prices in every round of the negotiation. Similarly to our strategy, they focused on the buyer side and they studied concurrent negotiations between a buyer and a set of sellers. In this setting, the buyer utilises a number of threads, with each thread following a specific strategy that adopts swarm intelligence techniques for achieving the optimal agreement. The PSO algorithm is adopted by each thread. What is interesting in their approach is that their strategy requires no central coordination. Their strategy does make some assumptions about the opponents. However, their experimental results focused on the time interval where an agreement is possible and were not compared with the current state-of-the-art (as we did). Also, Kolomvatsos et al. [34] used an artificial bee colony (ABC) algorithm to find the optimal agreement when negotiating with a group of sellers; however, the model has the same limitations as [33].

Williams et al. [63] proposed a negotiation strategy for concurrent negotiations in which the agent negotiates with multiple opponents about one item with multiple issues. This is the most similar existing research with respect to our setting. As with Nguyen and Jennings, the Williams et al. strategy has two components: a set of negotiation threads and a coordinator. Each negotiation thread is responsible for two tasks: (a) performing Gaussian process regression to predict the future utility of its opponent and (b) calculating the concession rate for an offer by taking into account the best time and best utility values from the coordinator. The coordinator is responsible for determining the best time and utility for each thread to reach an agreement. Williams et al. [63] developed two versions of their strategy. The first assumes that each thread has a different utility function from the others, while the second presupposes that all threads should have the same utility function.

However, there are weaknesses in the Williams et al. strategy: (a) both the agent and its opponents have the same deadline (and are aware of each other's), which is not realistic for most practical problems; (b) they model the openness of the market by having agents using the same break-off probability (likelihood of leaving the negotiation) which is prior knowledge that is not applicable in real e-markets where buyers and sellers are concurrently negotiating with each other; and (c) their negotiation setting is not completely open since they only allow agents to leave the environment, but not to enter during run-time. We will use the Williams et al. strategy to benchmark our strategy, as it represents the current state-of-the-art in multiple and concurrent bilateral negotiation.

Table 1 compares existing work in concurrent negotiation according to functionality: (a) continuous time—where real time is used during negotiation instead of using negotiation rounds or discrete time; (b) private deadline—where the buyer's and sellers' deadlines are unknown to each other in advance; (c) asynchronous negotiation—where the agent does not wait for all the sellers' counter-offers to reply; (d) environment model—where the buyer has a model of the market's environment in terms of the number of sellers, its competitors and other parameters that formulate the market's dynamics; (e) unknown opponents—where the buyer has no information about the strategy of the sellers; (f) incomplete information—where there is no a priory knowledge about the negotiation environment and the agents in it; (g) model of open market—where a negotiation agent is capable of modelling how agents enter and leave to/from the market at a time; (h) explicit decisions—where clear conditions about when to offer, accept, reqToReserve, cancel and exit are specified; (i) different price ranges—where the agent is tested against three classes of intersections between the price ranges (i.e. agreement zones) $[\mathrm{IP}_b, \mathrm{RP}_b]$ and $[\mathrm{IP}_{s_i}, \mathrm{RP}_{s_i}]$: 10, 60 and 100%; (j) opponent model—models the behaviour of the opponents; and (k) multi-issue negotiation—where buyers/sellers negotiate multiple issues like price and warranty.

**Table 1** Functionality comparison of the existing work in concurrent negotiation

| | Nguyen and Jennings [47] | An et al. [7] | Li et al. [39] | Kolomvatsos et al. [33] | Williams et al. [63] | CONAN |
|---|---|---|---|---|---|---|
| Continuous time | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| Private deadline | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Asynchronous negotiation | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Environment model | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| Unknown opponents | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ |
| Incomplete information | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ |
| Model of open market | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Explicit decisions | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Different price ranges | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Opponent model | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ |
| Multi-issue negotiation | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ |

## 2.2 Alternating offers protocol

We study a practical concurrent negotiation setting where a buyer agent engages in multiple bilateral negotiations in order to acquire a good. In doing so we experimented with the alternating offers protocol [54] due to its simplicity and wide use [37]. As shown in Fig. 2, in this protocol a buyer and a seller take turns to negotiate with possible actions: offer, accept and reject. The negotiation terminates either because an agreement is reached (buyer/seller accepts) or the negotiation has terminated (buyer/seller reaches their deadlines).

There have been two previous attempts to extend the alternating offers protocol. In [4], a *confirm* action is allowed after an *accept*; a buyer cannot cancel from confirmed offers. In [63], Fig. 3, a concurrent protocol is proposed and the possible actions are:

$A = \{\text{offer}[x], \text{accept}, \text{exit}, \text{confirm}, \textit{de-commit}\}$

The limitations in this method are that: (a) when the buyer makes a deal via a confirm action, the deal will be completed and the buyer will obtain the item under negotiation and the seller will receive the money for that item, so if the buyer cancels the deal (via a de-commit action), it results in a more complex situation where the buyer has to return the item back to the seller and the seller has to return the money back to the buyer, which can be inefficient in real-time applications; and (b) only the buyer can cancel the deal (via a de-commit action), which is not fair for the seller if the seller receives a better offer from other buyer. Moreover, the de-commit action is not explicitly shown in the state diagram of [62]. Furthermore, it implies that the protocol goes with a de-commit action from a final state (an agreement) to another final state (a conflict deal), which is logically problematic.

We will adopt the model of a percentage of the deal penalty to enable a fair comparison with the benchmark strategy of Williams et al. [63].

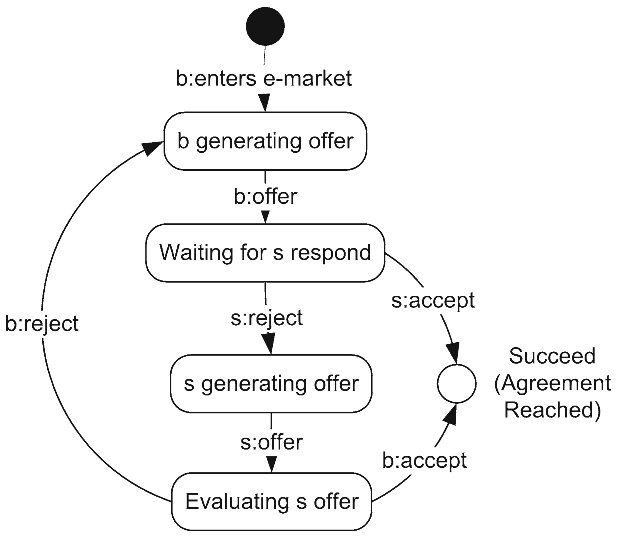


**Fig. 2** Alternating offers state diagram

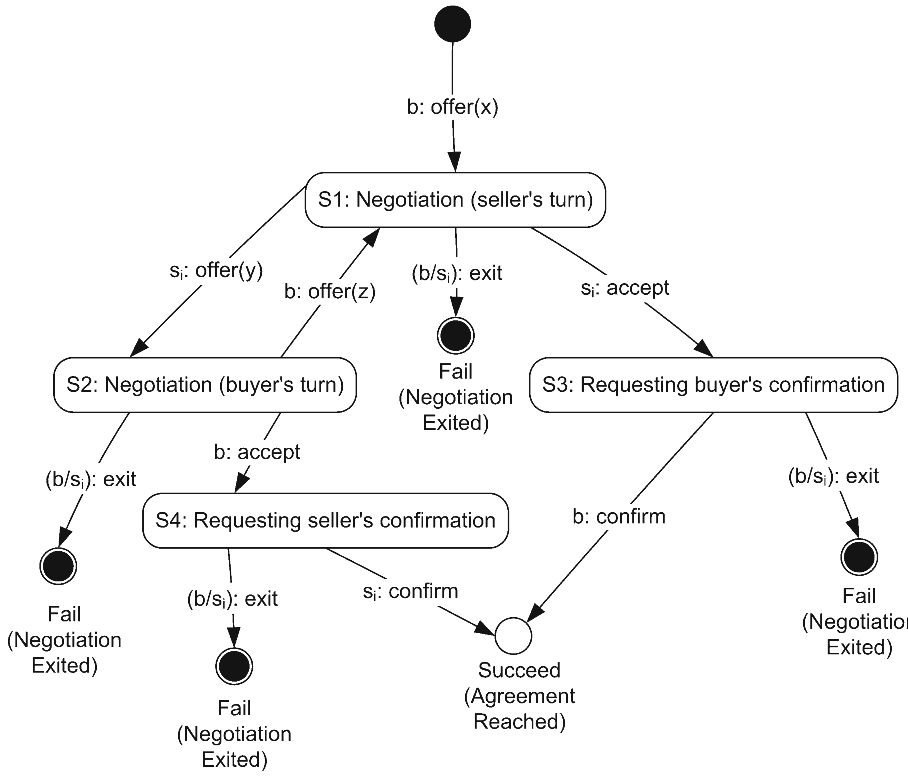

b: offer(x)
S1: Negotiation (seller's turn)
s_i: offer(y)
b: offer(z)
(b/s_i): exit
s_i: accept
Fail (Negotiation Exited)
S2: Negotiation (buyer's turn)
S3: Requesting buyer's confirmation
b: accept
(b/s_i): exit
S4: Requesting seller's confirmation
b: confirm
(b/s_i): exit
Fail (Negotiation Exited)
(b/s_i): exit
s_i: confirm
Fail (Negotiation Exited)
Succeed (Agreement Reached)
Fail (Negotiation Exited)

**Fig. 3** Williams' concurrent alternating offers state diagram

## 3 Concurrent negotiation setting

Given a domain of application such as buying and selling in an e-market place, automated negotiation is normally viewed as a mechanism [53] that relies upon two main elements:

- *Negotiation protocol* defines the space of possible agreements that agents can achieve and the legal actions that negotiating agents can make (i.e. the rules of encounter), including the process they need to follow to achieve these agreements. A protocol is public, viz., every agent knows the rules of negotiation in advance.
- *Negotiation strategy* determines an agent's behaviour during the negotiation process and specifies when and how to act. A strategy is private to the agent, i.e. we cannot assume that agents will know of the other agents' strategies.

A negotiating agent makes proposals defined by its protocol using its strategy. Negotiation proceeds as a series of rounds, and each round may involve one or more proposals. Negotiation terminates when agents reach an agreement, or a deadline is reached, whichever is earlier. Next, we describe the skeleton of our negotiation framework and the protocol used.

### 3.1 Skeleton of a concurrent negotiation

We assume that a market environment contains a set of buyers $B$ where $|B| = n$ $(n \geq 1)$ and a set of sellers $S = \{s_1, \ldots, s_m\}$ $(m \geq 1)$ that wish to negotiate over a set of resources $R = \{r_1, \ldots, r_l\}$ $(l \geq 1)$ using a protocol $p$. We develop a strategy for a buyer agent $b \in B$, who negotiates concurrently with multiple sellers over a resource $r \in R$ using the protocol $p$. Negotiation advances in real time $t$ $(t \in \mathbb{R}^+)$ where agents take decisions to make offers. At any time $t$ during a negotiation, we identify the following information.

$\text{Seller}_b^t$ is the set of seller agents negotiating with $b$ at time $t$, where $\text{Seller}_b^t \subseteq S$.

$C_b^t$ is the set of competitor agents for $b$ at time $t$, where $C_b^t \subseteq B \backslash \{b\}$.

The negotiation starts at time $T_s$. Once started, the negotiation can potentially last for time $T_b$, which is the maximum duration of time that $b$ can negotiate for. We also refer to $T_e$ as the deadline of negotiation for $b$, where $T_e = T_s + T_b$. We assume that $b$ does not know the sellers' deadlines, and that the sellers do not know $T_b$.

$b$ has $\left|\text{Seller}_b^t\right|$ negotiation threads, one for each seller $s_i \in \text{Seller}_b^t$.

$\delta_{ag_j \rightarrow ag_k}^t$ is a negotiation action from $ag_j$ communicated to $ag_k$ at time $t$, where either $(ag_j = b \wedge ag_k \in \text{Seller}_b^t)$ or $(ag_j \in \text{Seller}_b^t \wedge ag_k = b)$, and $\delta \in \{\mathsf{offer(x)}, \mathsf{reqToReserve}, \mathsf{reserve}, \mathsf{cancel}, \mathsf{confirm}, \mathsf{accept}, \mathsf{exit}\}$ where $\mathsf{offer(x)}$ is a term that contains the price $x$ of the resource $r$ under negotiation at time $t$.

$\text{IP}_b$ and $\text{RP}_b$ represent the initial and reservation prices of buyer $b$, while $\text{IP}_{s_i}$ and $\text{RP}_{s_i}$ represent the initial and reservation prices of seller $s_i$. In the literature, the intersection between the price ranges (i.e. agreement zones) $[\text{IP}_b, \text{RP}_b]$ and $[\text{RP}_{s_i}, \text{IP}_{s_i}]$ is always implicitly assumed to be 100% of $[\text{IP}_b, \text{RP}_b]$, a very difficult assumption to make when dealing with practical applications. So, in order to make our setting more realistic, we adopt three classes of intersections for the aforementioned price ranges: 10%, 60% and 100%. The reason behind this is that we want to capture situations where we can obtain higher utility, which was not previously possible without using negotiation. However, we do make the assumption that at any time $t$ if $\forall s_i \in \text{Seller}_b^t$ $(\text{IP}_b \geq \text{IP}_{s_i})$ holds, then there is no need to negotiate. This is a much softer assumption to make.

$\text{EG}_b \in [0, 1]$ represents the eagerness of buyer $b$, viz., specifying the willingness of $b$ to obtain a resource; this is specified by the user. For instance, if the user needs to buy a laptop to use it in his project in the next week, then the user has a high eagerness to buy a laptop.

On the other hand, if the user needs to move to a house in the next two years, then he has a low eagerness to buy a house.

$U_b(x) : \mathbb{R}^+ \rightarrow [0, 1]$ is the utility function that agent $b$ uses to determine the utility of an offer $x$ made either by $b$ or $s_i$ as follows:

$$U_b(x) = (\mathrm{RP}_b - x)/(\mathrm{RP}_b - \mathrm{IP}_b) \tag{1}$$

Since our focus is on a concurrent negotiation setting where a buyer agent $b$ engages in multiple bilateral negotiations with multiple sellers $s_i \in \mathrm{Seller}_b^t$ in order to acquire a single resource, we need a protocol that determines rules of negotiation that are appropriate for concurrent negotiation. The agent needs two types of protocols to rule the concurrent negotiation: the individual bilateral negotiation protocol and the overall negotiation protocol.

## 3.2 Bilateral protocol

In order to decide which protocol $p$ is allowed in our negotiation market, we experimented with the alternating offers protocol [54] because it is simple and widely used. However, this protocol is not sufficient to handle the complexity of concurrent negotiations due to its limited actions (i.e. offer(x), accept and reject). The following two cases show the difficulty in using the alternating offer protocol as is in a concurrent negotiation.

(i) While Bob is negotiating with Tom and about to send an accept to Tom's offer, a new seller John enters the market. This creates a new opportunity for Bob. As a result of using the alternating offers protocol, Bob needs to make a decision on how to proceed in the negotiation, either by: (a) buying from Tom without exploring what John has to offer; or (b) negotiating with John and thus risking to lose all exchanges between him and Tom that led Tom produce his current offer.

(ii) At time $t1$, Bob is negotiating concurrently with Alice, Tom and John. At time $t2$ Alice sends a counter-offer to Bob, while at time $t3$ Tom sends back an offer to Bob. At $t4$ Bob sends back a counter-offer to Alice and an accept to Tom. Also at $t4$ Bob receives an accept from John. Since Bob needs to buy one item only, the problem now arises at $t5$ of how will Bob deal with two accepts and whether Bob will continue the negotiation or not.

As shown in Fig. 4, we extend the protocol to allow for the additional actions: reqToReserve, reserve, cancel, confirm and exit. At negotiation time $t$, these actions can be described as follows.

- offer(x): represents an offer made either by $b$ (or $s_i$), proposing price $x$ for resource $r$;
- reqToReserve: provides $b$ with the opportunity to hold a preferred offer from a seller $s_i$ until a better offer is made by another seller $s_j$. The reason for reqToReserve actions is that $b$ cannot accept an offer directly because maybe a new seller will enter the market with a lower price. reqToReserve action is also useful because apart from helping to keep a good seller, there is a cancellation penalty which prevents the seller from exiting the negotiation. $b$ can send a reqToReserve to $s_i$ at any time $t$ if $b$ receives an accept from $s_i$ or $b$ receives an offer from $s_i$.
- reserve: gives $s_i$ with the opportunity to agree on holding a preferred offer from a buyer $b$. $s_i$ can send a reserve action to $b$ if $s_i$ receives a reqToReserve action from $b$.
- cancel: allows $b$ or $s_i$ or both to cancel from their reserved offers, which means that they retract or withdraw from existing reserved agreements. The cancelling agent pays a penalty *Penalty* to the other agent involved in the cancelled agreement, to ensure fairness and avoid unnecessary cancellation. We will discuss later how to deal with penalties.

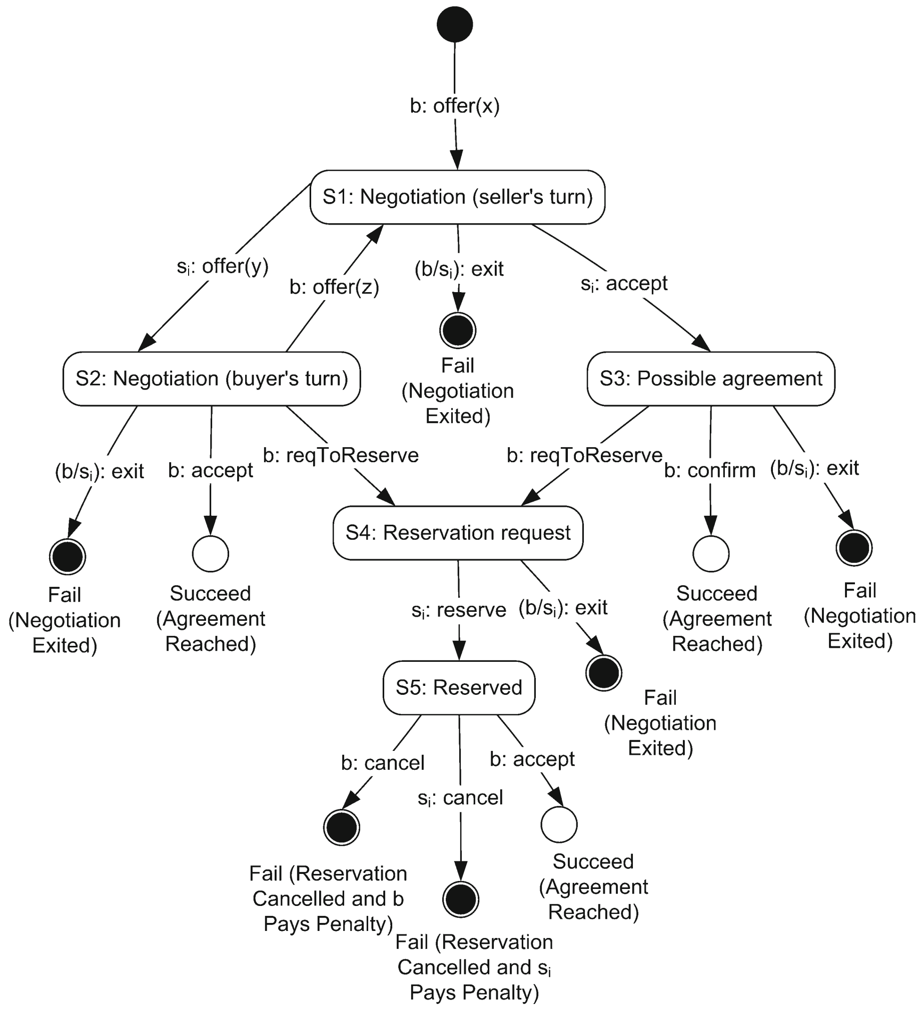

b: offer(x)
S1: Negotiation (seller's turn)
s_i: offer(y)
b: offer(z)
(b/s_i): exit
s_i: accept
Fail (Negotiation Exited)
S2: Negotiation (buyer's turn)
S3: Possible agreement
(b/s_i): exit
b: accept
b: reqToReserve
b: reqToReserve
b: confirm
(b/s_i): exit
S4: Reservation request
Fail (Negotiation Exited)
Succeed (Agreement Reached)
Succeed (Agreement Reached)
Fail (Negotiation Exited)
s_i: reserve
(b/s_i): exit
S5: Reserved
Fail (Negotiation Exited)
b: cancel
b: accept
s_i: cancel
Fail (Reservation Cancelled and b Pays Penalty)
Succeed (Agreement Reached)
Fail (Reservation Cancelled and s_i Pays Penalty)

**Fig. 4** CONAN concurrent alternating offers state diagram

– confirm: enables $b$ and $s_i$ reach an agreement about the resource $r$ under negotiation being sold at the price of the most current offer at time $t$. $b$ can send an confirm to $s_i$ at any time $t$ if $s_i$ accepts an offer from $b$.
– accept: permits $b$ and $s_i$ reach an agreement about the resource $r$ under negotiation being sold at the price of the most current offer at time $t$. $b$ can send an accept to $s_i$ at any time $t$ if: (i) $b$ reaches its deadline $t_b$ and has a reserved offer with $s_i$; and (ii) $b$ accepts an offer from $s_i$. In addition, $s_i$ can send an accept for any offer made by $b$.
– exit: supports $b$ or $s_i$ to withdraw from the course of negotiation at any time, even without notifying other agents. This implies that the negotiation between $b$ and $s_i$ has failed.

## 3.3 How To negotiate concurrently

Our protocol will be used by the buyer to communicate with multiple sellers concurrently. Negotiation starts when the buyer sends an offer to a seller and then waits for the seller's response. As soon as it receives a response from the seller, the buyer evaluates the seller's offer based on its strategy. If the buyer does not agree on the offer, it will send a counter-offer. The buyer and the seller continue to exchange offers until either agent exits the negotiation, an offer is reserved, or they reach an agreement. We assume that the agent has to reply to all pending offers and that messages received never expire. If the seller likes the buyer's offer it will send an accept action and wait for the buyer to:

- confirm the seller's accept, in which case the negotiation terminates with an agreement. The reason for this is to avoid the situation where *b* receives accepts from two different sellers;
- reqToReserve to the seller's accept, which implies that the buyer and seller move to the reservation request state and the buyer will continue to search for other better offers. However, the seller is not allowed to reqToReserve to the buyer's offer since we assume that the seller has large numbers of the same resource to sell; or
- exit from the negotiation with a seller and thus forcing negotiation to fail.

Likewise, if the buyer likes the seller's offer, it can either:

- accept the seller's offer and the negotiation terminates with an agreement; or
- reqToReserve to the seller's offer, so the buyer and seller move to the reservation request state and the buyer will continue to search for other better offers.

In the reservation request state the following may happen:

- the seller can exit from the reservation request state without paying any penalty. This move would be allowed only when the seller does not want to reserve the item and exits;
- the seller sends a reserve action agreeing to the reqToReserve from the buyer;
- the buyer or the seller cancel from the reserved state, which results in a penalty being paid by the participant who initiates the cancel; or
- the buyer can accept the reserved offer and the negotiation ends with an agreement.

## 3.4 Overall negotiation protocol

This protocol governs the communication between agents in the e-market. The overall negotiation starts when the agent *b* enters the e-market and finishes either when:

E1. *b* accepts or confirm an offer from a seller $s_i$; thus, agreement is reached and the negotiation succeeds; or
E2. *b* exits from all bilateral negotiations with all sellers; thus, no agreement was reached, *b* exits the e-market, and the negotiation fails.

An agent now needs a strategy to make decisions during negotiation. Such a strategy is protocol-dependent in that participating agents need to conform to the protocol's actions for progress to be made during negotiation. For this work we assume that agents are *protocol conformant*, i.e. that they will follow the protocol and will never breach it during negotiation.

## 3.5 Example run

To give an example of how agents interact with our overall negotiation protocol consider Fig. 5 depicting an example run using the protocol. The run shows the exchanges between agents for the scenario presented in the introduction. Each box corresponds to a bilateral negotiation between the buyer agent Bob and the seller agents Alice, Tom and John. Bob starts first with the two sellers Alice and Tom and, while he is negotiating with them, observes that a new

BILATERAL NEGOTIATION 1 (BOB AND ALICE):

Initial State $\xrightarrow{offer(700)^{t1}_{Bob\rightarrow Alice}}$ S1:Negotiation (seller's turn).

S1: Negotiation (seller's turn) $\xrightarrow{offer(950)^{t2}_{Alice\rightarrow Bob}}$ S2: Negotiation (buyer's turn).

S2: Negotiation (buyer's turn) $\xrightarrow{offer(750)^{t4}_{Bob\rightarrow Alice}}$ S1: Negotiation (seller's turn)

S1: Negotiation (seller's turn) $\xrightarrow{offer(850)^{t5}_{Alice\rightarrow Bob}}$ S2: Negotiation (buyer's turn).

S2: Negotiation (buyer's turn) $\xrightarrow{exit^{t6}_{Bob\rightarrow Alice}}$ Fail (Negotiation Exited).

BILATERAL NEGOTIATION 2 (BOB AND TOM):

Initial State $\xrightarrow{offer(700)^{t1}_{Bob\rightarrow Tom}}$ S1:Negotiation (seller's turn).

S1: Negotiation (seller's turn) $\xrightarrow{offer(800)^{t3}_{Tom\rightarrow Bob}}$ S2: Negotiation (buyer's turn).

S2: Negotiation (buyer's turn) $\xrightarrow{reqToReserve^{t4}_{Bob\rightarrow Tom}}$ S4: Reservation Request.

S4: Reservation request $\xrightarrow{reserve^{t5}_{Tom\rightarrow Bob}}$ S5: Reserved.

S5: Reserved $\xrightarrow{accept^{t6}_{Bob\rightarrow Tom}}$ Succeed (Agreement Reached).

BILATERAL NEGOTIATION 3 (BOB AND JOHN):

Initial State $\xrightarrow{offer(700)^{t4}_{Bob\rightarrow John}}$ S1:Negotiation (seller's turn).

S1: Negotiation (seller's turn) $\xrightarrow{offer(900)^{t5}_{John\rightarrow Bob}}$ S2: Negotiation (buyer's turn).

S2: Negotiation (buyer's turn) $\xrightarrow{exit^{t6}_{Bob\rightarrow John}}$ Fail (Negotiation Exited).

**Fig. 5** An example run of concurrent bilateral negotiation in CONAN

seller John is available, so he decides to negotiate with him too. This example shows that the difficulties that Bob had under the alternating offers protocol in Sect. 3.2, case (i), can be overcome mainly because Bob now is able to handle negotiations with Alice, Tom and the new seller John at the same time. More precisely, Bob can now send a reqToReserve to Tom to hold a good offer and at the same time start negotiating with John to explore what is it that John has to offer, thus improving his options to improve his utility during the negotiation. Eventually, the example shows that the bilateral negotiations of Bob with Alice and John fail, but Bob's negotiation with Tom succeeds. For a buyer like Bob, bilateral negotiations are separate concurrent threads that encapsulate their own initial states and form part of the buyer's overall internal state. Interactions within threads are viewed as state transitions that follow the protocol of Fig. 4. $S1, S2, \ldots S6$ are state labels for the state of the buyer. The state transitions of the buyer are then caused by negotiation actions of the form $\delta^{t}_{ag_j\rightarrow ag_k}$ (as discussed earlier).

## 3.6 Implementation

We build a negotiation framework using RECON [2], an experimental simulation platform that supports the development of software agents interacting concurrently with other agents in negotiation domains. Unlike existing simulation toolkits [42], RECON also supports declarative strategies to enable agents explain their decisions to a user, if necessary. Our buyer agent is implemented according to KGP agent model [57] with the specialised agent architecture discussed in [3]. RECON is built on top of GOLEM [1] [12] suitably specialised with infrastructure agents to manage an e-market and extract statistics from the negotiations that take place. A tutorial video on RECON is available online.[2]

## 4 CONAN

We study the development of a heuristic negotiation strategy for a buyer agent in a concurrent negotiation market setting. Existing work [47,63] provides answers only about when to offer or accept and what to offer, but does not explicitly represent reqToReserve and cancel actions in a concurrent negotiation dialogue. In addition, offers are computed using an opponent's model and do not take into account the environment and/or self models [47,63]. Moreover, some existing work assumes complete and certain information about the negotiation environment and the opponents [47]. In most approaches the agent strategy is an isolated component without clear illustration of how the model is designed and how different components interact with each other [47,63]. To develop an adaptive negotiation decision-making model in a dynamic environment, we must support an agent to deal with an enormous number of environmental and attitude changes at the same time. To address this issue, we develop the CONAN strategy. Since our buyer agent needs to acquire a resource when negotiating with multiple sellers at the same time, the agent will use CONAN to decide (1) what actions to take during negotiation and when to take them; and (2) whether the action is an offer, then how to generate an offer. Algorithm 1 gives an overview of CONAN.

**Algorithm 1:** Reaching agreements via concurrent negotiation

```
   Input: $T_b$: negotiation duration,
   $\mathrm{IP}_b$: buyer's initial price,
   $\mathrm{RP}_b$: buyer's reservation price.
   Output: offer (offer $>$ 0 if agreement, offer = 0 if no agreement)
1  $T_e = T_s + T_b$
2  Repeat
3      Get active sellers $\mathrm{Seller}_b^t$
4      Calculate the $\mathrm{CR}_{t,s_i}$ for each seller $s_i \in \mathrm{Seller}_b^t$
5      Generate $b$'s next offer for each $s_i$
6      Compute $\delta_{b\to s_i}^t = \{exit, reqToReserve, cancel, accept, confirm, offer\}$.
7      Send $\delta_{b\to s_i}^t$ for each $s_i$
8  Until $t = T_e$ || $\delta_{b\to s_i}^t = accept$ || $\delta_{b\to s_i}^t = confirm$
9  if $\delta_{b\to s_i}^t = accept$ || $\delta_{b\to s_i}^t = confirm$ then
10     Return offer (agreement)
11 else
12     Return 0 (no agreement)
```

CONAN takes $T_b$, $\mathrm{IP}_b$ and $\mathrm{RP}_b$ as an input from the agent user and returns either an agreement or no agreement to allocate a resource as output. CONAN starts by initialising its deadline, then calculates the concession rate $\mathrm{CR}_{t,s_i}$, generates the next offer and decides

1 http://golem.cs.rhul.ac.uk/.

2 https://youtu.be/PmuHhxgFqpc.

which action to proceed with for each seller $s_i$. The designs of each heuristic are discussed in sub-sections below.

## 4.1 Heuristics to calculate the concession rate

CONAN generates offers $\text{offer}_{t,s_i}$ at time $t$ for seller $s_i$ based on the following equation [18]:

$$\text{offer}_{t,s_i} = \text{IP}_b + (\text{RP}_b - \text{IP}_b) * \text{CR}_{t,s_i} \tag{2}$$

where

- $\text{IP}_b$: is the initial price;
- $\text{RP}_b$: is the reservation price;
- $\text{CR}_{t,s_i}$: is the concession rate at time $t$ for seller $s_i$. $\text{CR}_{t,s_i} \in [0, 1]$ and $\text{offer}_{t,s_i} \in [\text{IP}_b, \text{RP}_b]$.

Our evaluation of the concession rate depends on the environment and self factors:

$$\text{CR}_{t,s_i} = \begin{cases} 0, & \text{if } t = T_s; \\ 0.99, & \text{if } t = T_e - \alpha \text{ where } \alpha = \text{ time to complete one} \\ & \text{negotiation round;} \\ \text{CR}_{t-1,s_i}, & \text{if } \text{CR}_{t,s_i} - \text{CR}_{t-1,s_i} < 0; \\ \text{CR}_{t-1,s_j} * 0.9, & \text{if offer of } \text{CR}_{t-1,s_j} \text{ reserved and } j \neq i; \\ w_{\text{Env}_t} E_t + w_{\text{Self}_t} S_t, & \text{otherwise} \end{cases} \tag{3}$$

where

- $E_t$: is the effect of the environment factor where $E_t \in [0, 1]$;
- $S_t$: is the effect of self factor where $S_t \in [0, 1]$;
- $w_{\text{Env}_t}, w_{\text{Self}_t}$: are the environment and self weights, normalised in the following sense: $w_{\text{Self}_t}, w_{\text{Env}_t} \in [0, 1]$ and $w_{\text{Self}_t} + w_{\text{Env}_t} = 1$.

Before explaining the environment and self factors $E_t$ and $S_t$, we demonstrate how we calculate each case of the concession rate $\text{CR}_{t,s_i}$ in Eq. 3 above.

**Case 1** When $b$ starts the negotiation, it will offer its initial price first.
**Case 2** When time is $\alpha$ time units close to the deadline and the buyer $b$ has no reserved offer, then it needs to secure an offer even if it is near to its reservation price. Therefore, $b$ concedes near to its reservation price by setting the concession rate $\text{CR}_{t,s_i} = 0.99$.
**Case 3** When $b$ generates an offer that is non-monotonically increasing (i.e. $\text{CR}_{t,s_i} - \text{CR}_{t-1,s_i} < 0$) at $t$, then, to ensure its own rationality, $b$ will actually offer its previous offer $\text{CR}_{t-1,s_i}$. This case needs a first generation of the value $\text{CR}_{t,s_i}$ with cases 1 or 5.
**Case 4** If $b$ has a reserved offer at $t - 1$, there is no need for $b$ to concede more to other sellers. Thus, $b$ will generate an offer which is less than 10% of its reserved offer. This case needs a first generation of the value $\text{CR}_{t,s_i}$ with cases 1 or 5.
**Case 5** Otherwise, $b$ will generate an offer that takes into consideration the environment and self factors. To give a flavour of these, the following are some special cases illustrating how environment and self factors affect the concession rate.

(a) Adding and/or removing a seller to/from the negotiation: while agent $b$ is negotiating with agent $s1$ and agent $s2$, a new seller agent $s3$ enters the market and starts a new negotiation thread with $b$. This creates a new option for $b$. As a result, $b$ will decrease its concession rate to see how the negotiation process proceeds with agent $s3$. On the other hand, when $s1$ exits the negotiation, it means $b$ loses an option; thus, it has to increase its concession.

(b) Adding and/or removing negotiation competitors: consider agent $b$ negotiating with agents $s1$ and $s2$ as before, but now $b$ perceives that a competitor $c1$ is negotiating with agent $s1$. Hence, $b$ may concede by a larger amount to attract $s1$ to sell the resource. In contrast, if $c1$ leaves the market, $b$ may concede by a smaller amount since it has no other competitors while negotiating with $s1$.

(c) Change in the negotiation situation: if $b$ is in a bad negotiation situation (a notion to be defined by a function when we explain the self factor $S_t$), then $b$ has to concede more in order to secure the resource before the deadline. On the contrary, when in a good negotiation situation, $b$ has a higher chance of winning the negotiation, which in turn leads to $b$ reducing its concession rate.

Now, we will explain the self and environment factors $S_t$ and $E_t$.

### *4.1.1 Self factor*

Given a concurrent negotiation setting, the self factor is an indication of how well the agent is doing at a specific time. In CONAN, we address this point heuristically by studying a self factor $S_t \in [0, 1]$ that depends on how many reserved offers (RO) the agent has made so far, the agent's negotiation situation (NS) representing progress with all negotiation threads (i.e. how well opponents respond to our offers and how opponents concede over time), the effect of the passage of time (TE) and the eagerness ($\text{EG}_b$) of our buyer agent to obtain the item under negotiation. We compute the self factor as:

$$S_t = \left(\frac{1}{\text{RO}+1} + \text{NS} + \text{TE} + \text{EG}_b\right)/4 \tag{4}$$

As we expect each sub-factor RO, NS, TE, $\text{EG}_b \in [0, 1]$, we divide the final sum by 4 to ensure that the value for $S_t \in [0, 1]$. We are now in the position to explain each sub-factor in detail.

RO: is the number of reserved offers, obtained from the set of reserved offers $\text{RO}_t$ at time $t$ and evaluated as $|\text{RO}_t|$. $\text{RO}_t$ (and as result RO) is updated every time the buyer reserves an offer or cancels a reservation. If the number of reserved offers increases, then the agent decreases the concession rate for generated offers. We normalise RO $\in [0, 1]$.
NS: is the negotiation situation for all threads. The evaluation of the negotiation situation in each thread is used to decide the next offer for each thread. It is calculated using the following two criteria: *Opponent's response time* and *Concession rate*. These criteria are based on combinations of time-dependent factors (for the opponent's response time) and behaviour-dependent factors (for the opponent's concession rate), which are inspired by the works described in [18,20].

**Opponent's response time** is the time taken for the opponent to make a counter-offer given the time the buyer $b$ has available to negotiate. More specifically, given an offer from the buyer $b$ to seller $s_i$ at time $t_l$, denoted as $\delta^{t_l}_{b\rightarrow s_i}$, and a response to that offer from $s_i$ to $b$ at time $t_m$ ($t_m > t_l$), denoted as $\delta^{t_m}_{s_i\rightarrow b}$, then *Opponent's response time* is as described below:

$$\textit{Opponent's response time} = (t_m - t_l)/T_b \tag{5}$$

*Opponent's response time* $\in (0, 1]$. The actual value for the response time of an opponent during negotiation classifies opponents into three distinct classes: *compatible*, *moderately compatible* or *incompatible*. Each opponent class is obtained by dividing the possible values of the *Opponent's response time* (0, 1] into three equal sub-intervals shown below:

**Table 2** Mapping procedure using the Borda method

| Range | Qualitative decision | Score |
|---|---|---|
| $(0, \frac{1}{3}]$ | Compatible | 1 |
| $(\frac{1}{3}, \frac{2}{3}]$ | Moderately compatible | 2 |
| $(\frac{2}{3}, 1]$ | Incompatible | 3 |

- compatible $\in (0, \frac{1}{3}]$;
- moderately compatible $\in (\frac{1}{3}, \frac{2}{3}]$;
- incompatible $\in (\frac{2}{3}, 1]$.

The smaller the response time, the more compatible an opponent is. These three notions of compatibility are used here as a heuristic way to classify the behaviour of opponent responses in relation to the expectations of the buyer agent. The intuition is that this classification is sufficiently discriminating if compared with the bipolar alternative compatible/incompatible, while at the same time it is equally simple to explain and has the added advantages of distinguishing behaviours that lie in the middle.

**Opponent's concession rate** is the rate of change between two consecutive offers of the opponent at time $t = t_k$ over the price range the agent has to negotiate. This is based on a window of consecutive opponent offers $\phi > 1$, containing first the last offer made to $b$ by $s_i$, denoted as $\text{offer}_{s_i \to b}^{t_{k-1}}$, and ending with $\text{offer}_{s_i \to b}^{t_{k-\phi}}$. We can then compute the opponent's concession rate as follows:

$$Opponent's\, concession\, rate\ (\phi) = 1 - \left( \frac{\text{offer}_{s_i \to b}^{t_{k-\phi}} - \text{offer}_{s_i \to b}^{t_{k-1}}}{\text{RP}_b - \text{IP}_b} \right) \tag{6}$$

For instance, if $\phi = 3$ and the previous consecutive opponent offers of $s_i$ to $b$ at time $t_k$ are 770, 790 and 800, then the concession will be $(800 - 770)/3$. We then normalise the *Opponent's concession rate* to a real number $\in [0, 1]$ and map the result to incompatible, moderately compatible and compatible, as with the *Opponent's response time*.

From these two criteria, our goal is to derive a *local situation* $\sigma_i$ for each negotiation thread $i$ (where $i$ represents the negotiation of buyer $b$ with seller $s_i$), which also takes the values incompatible, moderately compatible and compatible. To compute $\sigma_i$, we have chosen the multi-criteria decision-making process known as the Borda method [11], giving a value to every qualitative decision based on its position, as in Table 2. We select this method because it: (1) supports proportional representation of the situation of each thread; (2) is neutral and monotonic [10]; (3) produces a complete, transitive ranking for a set of alternatives [17]; (4) preserves the decisiveness of labels (incompatible, moderately compatible and compatible) [25]; (5) uses straightforward calculation; and (6) employs feasible computation.

Then $\sigma_i$ sums up the scores *Opponent's response time* and *Opponent's concession rate*:

$$\sigma_i = Opponent's\, response\, time + Opponent's\, concession\, rate. \tag{7}$$

Given the score combinations of *Opponent's response time* and *Opponent's concession rate*, the domain of $\sigma_i$ is [2, 6], with 2 implying that the opponent is compatible with our style of negotiation and 6 implying that the opponent is incompatible. We then derive the global negotiation situation $\sigma$ for buyer $b$, which we calculate as the sum of all the local negotiation situations $\sigma_i$, as shown in Eq. (8). However, $\sigma$ needs to be normalised, so that we can use the normalised value to generate the concession rate. So NS is the normalised global negotiation situation given by Eq. 9, where 2 and 6 are the minimum and maximum values that a thread can score, respectively, while $y$ is the number of sellers $y = |\mathrm{Seller}_b^t|$.

$$\sigma = \sum_{i=1}^{y} \sigma_i \tag{8}$$

$$\mathrm{NS} = \frac{\sigma - (2 * y)}{(6 * y) - (2 * y)} = \frac{\sigma - (2 * y)}{4 * y} \tag{9}$$

TE: is the effect of the passage of time on the concession rate for buyer $b$. We use Eq. 10 to normalise TE between [0, 1].

$$\mathrm{TE} = \frac{t - T_s}{T_b} \tag{10}$$

$\mathrm{EG}_b$: is the eagerness of buyer $b$ to obtain the resource and is specified by the user. Eagerness means the degree of desirability of an item by the buyer. In practice, if the eagerness of the buyer for an item sold by a seller is higher than that sold by another, then the buyer is willing to concede more on the price of the more desired item, not only because of the deadline. As discussed in Sect. 3, $\mathrm{EG}_b$ is between [0, 1].

### *4.1.2 Environment factor*

The environment factor allows a buyer agent using CONAN to answer the question of how suitable is the environment to the agent at a given time. As with the self factor, the environment factor is a heuristic originating from the social sciences based on the negotiation power principle, which gives more power to the buyer to explore different alternatives when negotiating with many opponents [22]. Given a dynamic market environment, the environment factor $E_t \in [0, 1]$ depends on the number of sellers ($\mathrm{Se}_t$) that are available, the number of competitor agents ($C_t$) that are after our item and the demand/supply ratio ($R_{\mathrm{ds}}$). We compute the environment factor as:

$$E_t = \left(\frac{1}{\mathrm{Se}_t} + C_t + R_{\mathrm{ds}}\right)/3 \tag{11}$$

As we expect the sub-factors $\mathrm{Se}_t$, $C_t$, $R_{\mathrm{ds}} \in [0, 1]$, then we need to divide the sum of all sub-factors by 3 to obtain a value for $E_t \in [0, 1]$. We explain next each sub-factor.

$\mathrm{Se}_t$: is the number of sellers that are actively negotiating with $b$. The value of $1/\mathrm{Se}_t \in (0, 1]$ (see case 5(a)).

$C_t$: is the number of active competitor agents, other buyers who are trying to make a deal with the sellers negotiating with $b$ for the same resource. We assume that this number is obtained from the market. The lower the number of competitors during negotiations, the higher the possibility of $b$ reaching an agreement by conceding less [see case 5(b)]. We normalise the value of $C_t$ between [0, 1].

$R_{\mathrm{ds}}$: is the demand/supply ratio calculated as the ratio of the number of buyers to the number of sellers. The higher the ratio, the higher the price for the resource and, hence, the more difficult it is to reach an agreement. There are many e-markets in the literature that provide demand/supply ratios [55], including Tete-a-Tete, Kasbah, AuctionBot and the Fisher market. If the e-market does not provide the agents with the demand/supply ratio, then it can be calculated by dividing the number of sellers by the number of competitors. For instance, E-Bay provides the user with the number of buyers that follow specific item (number of competitors). We normalise the value of $R_{\mathrm{ds}}$ between [0, 1].

### *4.1.3 Assigning weights to factors*

We develop a method to assign weights to the environment and self factors presented earlier. First, in Eq. 12, we assign a weight to the self factor $S_t$ depending on the value of the factor itself. Then, in Eq. 13, we calculate the impact of the price range DM. The functions below present this method. We classify the value of $S_t$ as low, medium or high, where:

- low $\in (0, \frac{1}{3}]$;
- medium $\in (\frac{1}{3}, \frac{2}{3}]$;
- high $\in (\frac{2}{3}, 1]$.

The reason for the classification as low, medium or high is the same as that for choosing to classify as incompatible, moderately compatible or compatible in Sect. 4.1.1. The buyer always looks for its self factor $S_t$ first, to see how satisfied it is with the current progress of the negotiation. If the value of $S_t$ is low, it means that the buyer only by looking on this factor will concede less, which implies that the buyer is in a good negotiation position; focusing on $S_t$ only, $b$ will concede only by a small amount. For this reason, $b$ must put more weight on the self factor, as it suggests that $b$ should concede very little, which in turn implies that the environment factor should be assigned a lower weight. As a result, a high weight will be assigned to $w_{\text{Self}_t}$. This is captured by Eq. 12, where if the value of $S_t$ is low, we multiply by 0.75 to make it more important; if the value of $S_t$ is medium, by 0.5 to make it indifferent; and if the value of $S_t$ is high, by 0.25 to make it less important. In this way, $w_{\text{Self}_t}$ is adjusted according to the negotiation progress.

$$w_{\text{Self}_t} = \begin{cases} \text{DM} * 0.75, & \text{if } S_t = \text{low} \\ \text{DM} * 0.5, & \text{if } S_t = \text{medium} \\ \text{DM} * 0.25, & \text{if } S_t = \text{high} \end{cases} \tag{12}$$

We also need to cater for the situation where the initial price of the seller is far away from the reservation price of the buyer. In this case, even if $S_t$ is low, $b$ will need to concede more, in order to close the price gap in fewer negotiation cycles. We therefore introduce a price distance multiplier DM in Eq. 12, which measures the negotiation gap between $b$ and $s_i$ and is defined by Eq. 13.

$$\text{DM} = (\text{IP}_{s_i}/\text{RP}_b) * \text{TE} \tag{13}$$

The ratio estimated by the initial price of the seller $\text{IP}_{s_i}$ and the reservation price of the buyer $R_b$ is multiplied by TE, which is the effect of time passage, to remove a small percentage from the distance. The domain of TE is within the interval [0, 1]. We are now in a position to evaluate the weight of the environment factor, as shown in Eq. 14.

$$w_{\text{Env}_t} = 1 - w_{\text{Self}_t} \tag{14}$$

## 4.2 Cancellation penalty

The cancellation penalty is important to ensure fairness between negotiation participants, to save money and resources and to allow the system to be stable in terms of avoiding unwanted cancelled deals.

The penalty amount needs to be determined by the agent's user before the start of the negotiation in order to decide when to cancel from an offer. $\text{Penalty}_b$ represents the total amount of penalties for all the reserved offers. Thus, if the agent, during negotiation, finds a better offer, then the agent will cancel from the least preferred offer (i.e. one cancel only) and reqToReserve to the better offer. At the end of the deadline, the buyer accepts its most preferred reserved offer and cancels the rest of the reserved offers. The buyer keeps an MRO (maximum number of reserved offers), which when it is reached the buyer cannot reserve any further offer.

The penalty the buyer has to pay $\text{Penalty}_b$ is calculated as the difference between the amount of money the buyer has to pay to cancel any reserved offers and the amount of money the buyer has already received from opponents that cancelled their reservations with the buyer, i.e.:

$$\text{Penalty}_{b,s_i} = \text{Penalty}_{s_i,b} = D * \text{Price}_{\text{reserved}} \tag{15}$$

$$\text{Penalty}_b = (\text{Penalty}_{b,s_i} * \text{RO}) - (\text{Penalty}_{s_i,b} * \text{RO}_{\text{Seller}_b^t}) \tag{16}$$

where $\text{Penalty}_{b,s_i}$ is the penalty paid by $b$ to cancel an offer made to seller $s_i$, $\text{Penalty}_{s_i,b}$ is the penalty paid by $s_i$ to cancel an offer made by $b$, $D$ is the percentage to be paid from the cancelled offer $\in [0-1]$, RO is the number of reserved offers so far, and $\text{RO}_{\text{Seller}_b^t}$ is the number of cancelled offers that occur from sellers $\text{Seller}_b^t$ negotiating with $b$.

The reason for choosing $D$ (a percentage of the reserved offer) is for fair comparison with the benchmark, viz., the strategy used by Williams et al. [63].

## 4.3 Heuristics to decide actions

In order for the buyer $b$ to decide which action to choose at each point in time, we developed a rule-based strategy. We start with the reqToReserve and cancel and then discuss the accept conditions. In these rules, we will use three parameters determined by the user: (a) MRO—this is the maximum number of reserved offers; (b) $D$—this is the percentage of the deal for cancelling from negotiation threads (more details in Sect. 4.2); and (c) MAN—this is the maximum number of active threads (ensures that the agent has limited computational resources).

### *4.3.1 Conditions to request to reserve*

Agent $b$ reqToReserve to $s_i$ when:

- offer $\delta^t_{s_i \to b} + \text{Penalty}_b \leq \delta^{t+1}_{b \to s_i} <$ any reserved offers and $b$ is not near the deadline.
- $s_i$ accepts the last offer from $b$ and $b$'s last proposed offer + $\text{Penalty}_b \leq$ any reserved offers and $b$ is not near the deadline.

### *4.3.2 Conditions to cancel*

Agent $b$ cancels from:

- the least preferred offer when RO $\geq$ MRO;
- all reserved offers immediately after $b$ has accepted an offer.

### *4.3.3 Conditions to accept*

Agent $b$ accepts:

- $\delta^t_{s_i \to b}$ when offer $\delta^t_{s_i \to b} + \text{Penalty}_b \leq \delta^{t+1}_{b \to s_i} <$ any reserved offers and $b$ is nearing the deadline.
- $s_i$'s acceptance when $s_i$ accepted the last offer from $b$ and $b$'s last proposed offer + $\text{Penalty}_b \leq$ any reserved offers and $b$ is nearing the deadline.
- the highest utility reserved offer when $b$ is nearing the deadline $t = T_e$.

### *4.3.4 Conditions to confirm*

Agent $b$ confirm:

- $s_i$'s acceptance when $s_i$ accepted the last offer from $b$ and $b$'s last proposed offer + $\text{Penalty}_b \leq$ any reserved offers and $b$ is nearing the deadline.

### *4.3.5 Conditions to exit*

We study next the conditions that the buyer $b$ needs to check before exiting from an individual thread or from the negotiation as a whole.

**Conditions to exit from a thread**

Agent $b$ exits from:

- any thread if the deadline has been reached, i.e. $t = T_e$;
- $\sigma_i$ if seller $s_i$ is incompatible (i.e. based on the evaluation NS mentioned above) and there is a new seller $s_j$ and $|\text{Seller}_b^t| = \text{MAN}$ (which means that the current opponent is incompatible and the buyer has a new opponent to negotiate with, but due to the limited number of active negotiations, the agent has to terminate the negotiation with the incompatible opponent).
- $\sigma_i$ if $s_i$ has accepted an offer, but a higher utility offer has been accepted at the same time by $s_j$ (which means that the buyer receives two accept actions from two different sellers at the same time, leading the buyer to choose the highest utility offer from $s_j$ and exit from $s_i$).

**Conditions to exit from the whole negotiation**

Agent $b$ exits from the whole of the negotiation

- if the deadline has been reached, i.e. $t = T_e$.
- if $\delta^t_{b \to s_i}$ = accept.

## 4.4 Representation of the strategy

In the context of our implementation discussed in Sect. 3.6, we represent the state of an agent during negotiation as a temporal logic program implemented in Prolog, which we assume the reader is familiar with. Prolog uses the convention that a constant is an identifier starting with a lower-case letter, while a variable is an identifier starting with a upper-case letter. A constant stands for a specific entity, and different constants stand for different entities. On the other hand, a variable can stand for any entity, and different variables can stand for the same entity. The predicate name must be a constant, while each argument can either be a constant or a variable. A Prolog program can be used to answer queries, or to achieve goals, the prompt "?-" denotes a query whose truth will be evaluated by the program.

The temporal part of the logic is represented using the Event Calculus (EC) [35], a logical language for reasoning about actions and their effects. We will use a dialect of the EC based on temporal variables known as multi-valued fluents due to the fact that these variables can take different values at different times. At a time T a fluent is represented in the form F=V to denote that it holds at T that F has value V [8]. For instance, to query that the initial price of specific laptop, say `laptop12`, costs `1200` at the current time `15`, we write:

```
?- holds_at(initial_price(laptop12) = 1200, 15).
```

The state of the agent will generally contain many other fluents to represent information about negotiation threads, information about other agents and knowledge about the environment.

From a representation of state such as the one above, we develop a strategy as an action selection policy whose specification is defined by Prolog rules of the form:

```
select(Goal, Action, T):- Conditions [T].
```

An example of a part of the strategy is shown in Listing 1, showing how the buyer *b* can reason about which action to select next.

```
 1  select(buyfrom(Si, Item, ao), offer(Si, Item, Offer), T):-
 2       deadline(Deadline),
 3       ourdeadline(StartTime),
 4       Td is (StartTime + Deadline),
 5       T < Td,
 6       calc_next_offer(buyfrom(Si, Item, ao), Offer, T).

 8  calc_next_offer(buy(Si, Item, Ao), Offer, T):-
 9       concession(buy(Si, Item, Ao), CA, T),
10       Offer is Min + (Max - Min) * CA.
```

**Listing 1** Extract from the declarative specification of the agent strategy in Prolog

Listing 1 assumes that the buyer has a goal to buy an item from the market, and has to identify a seller `Si`, and create a thread of negotiation with that seller. This is done by the buyer creating a sub-goal to buy the `Item` from `Si` using the concurrent alternating offers protocol `ao`. The predicate representing the head of the rule

```
select(buyfrom(Si, Item, ao), offer(Si, Item, Offer), T)
```

(line 1) returns an offer (represented by the variable `Offer`) for the agent's goal (`buyfrom(Si, Item, ao)`), seller (`Si`) and item under negotiation (`Item`). This predicate calls another predicate that acts as a function to generate the counter-offer at line 6 after checking the buyer's deadline. The predicate call

```
calc_next_offer(buy(Si, Item, Ao), Offer, T)
```

(line 8) generates the counteroffer at time `T` for seller (`Si`) based on the initial price (`Min`), reservation price (`Max`) and concession rate (`CA`). The predicate call

```
concession(buy(Si, Item, Ao), CA,  T)
```

(line 9) calculates the concession rate (`CA`) at time `T` for seller (`Si`) according to Eq. 3. The remaining of the strategy is defined with similar rules that are called from within a declarative agent cycle. Details on how the strategy is incorporated in the buyer's agent cycle are beyond the scope of this paper and the interested reader is referred to our previous work [3], where this issue has been discussed.

## 5 Empirical evaluation

In the previous sections we have presented the background that motivated the development of CONAN and the model itself. This section presents the experimental results of the work and their evaluation. We proceed as follows. We outline first our negotiation simulation framework RECON [2], which implements the negotiation agents and facilitates the way we have generated our experimental results. We present two large experiments to test CONAN. We report their results and evaluate their significance.

## 5.1 Experiment 1: comparing CONAN with the state-of-the-art

The aim of this experiment is to determine whether CONAN can outperform the state-of-the-art and if so, under what settings. This section starts by explaining the experimental setting in detail, which includes the choice of opponents, the benchmark strategy that we are comparing our work with, the parameters of the proposed experiments, and the performance measurements characterising our results. After that, we report on the experimental results.

### *5.1.1 E-market setting*

As discussed in Sect. 3.6, we implemented CONAN agents using the architecture proposed in [3]. Interaction between CONAN and other agents is handled using inter-agent communication. As explained in Sect. 3, RECON supports both the imperative definition of agent strategies in Java and the declarative specification of logic-based strategies in Prolog, within the same environment. This feature is important, because we can develop simulations with CONAN developed declaratively and then compete with imperative agents, as those currently representing the state-of-the-art in the negotiation literature, without any difficulties. In addition, RECON supports concurrent negotiation which, at the time of our experimentation effort, other existing negotiation platforms did not support [42].

**Choice of opponents** For our evaluation, we developed extended versions of Faratin's [18] and ANAC 2011 (International Automated Negotiating Agents Competition) [24] strategies that allow opponents (sellers) to concurrently negotiate with different buyers. In the simulation, we divide these strategies into three groups:

– The first group, *Faratin's Sellers* [18], contains three combinations of strategies which we combine into one group. The first combination is called time-dependent strategies, where the value of the counter-offers and the acceptance value for the offers depend on the remaining negotiation time. The second and third combinations are the resource-dependent and behaviour-dependent strategies. In resource-dependent strategies, sellers concede more rapidly as the quantity of resources becomes limited. As in Faratin et al. [18], resources are the number of buyers participating in the negotiation and the negotiation time. In behaviour-dependent strategies, the seller imitates the behaviour of the buyers, so the seller will compute the next offer based on the previous attitude of the buyer.
– The second group is the *ANAC Sellers* [24]. We chose this group for fair comparison with the benchmark strategy, which we will describe once we have completed this discussion on opponents. In addition, the ANAC competition provides the most recent and most competitive state-of-the-art practical negotiation agents. It is important to note that in our experiments we noticed, in the 10% agreement zone (for more details see Sect. 3), that three ANAC sellers were accepting offers with negative utility for them. This was due to the fact that ANAC sellers always assume that there will be a 100% agreement zone between them and the buyers. To make the comparison fair with ANAC sellers in the 10% agreement zone, we adjusted their strategies to offer or accept only when they obtained positive utility.
– The third group, *All Sellers*, is a combination of all *Faratin's Sellers* and *ANAC Sellers*.

Each seller has two fixed private deadlines. The first is the deadline for how long it will be staying in the market. The second is the deadline related to the negotiation with each buyer. When a seller starts to negotiate with each buyer, it assigns a fixed duration for the

**Table 3** Simulation parameters

| Variables | Description |
|---|---|
| Buyer's initial price | Starting price $\mathrm{IP}_b$ of buyer $b$ |
| Buyer's reservation price | Maximum price $\mathrm{RP}_b$ buyer $b$ will offer |
| Seller's initial price | Starting price $\mathrm{IP}_{s_i}$ of seller $s_i$ |
| Seller's reservation price | Maximum price $\mathrm{RP}_{s_i}$ seller $s_i$ will offer |
| Market update time | Elapsed time for the market to change (population and demand / supply ratio due to new agents entering or existing agents leaving the market) |
| Change in market population | The rate in which the number of agents increases or decreases |
| Change in demand/supply | The rate in which the ratio of buyers to sellers increases or decreases |
| Deadline | Duration of the negotiation $T_b$ for buyer $b$ |
| Eagerness | The eagerness $Eg_b$ of buyer $b$ |

negotiation, for instance, three minutes. Hence, when the seller starts a new negotiation with another buyer, it negotiates for a fixed time, three minutes only. The seller will exit the market if (a) the seller sells all its resources, or (b) the seller reaches the deadline for staying in the market, or (c) the market forces the sellers to exit their negotiations to alter the demand/supply ratio of the market.

**Benchmark strategy** We will use the state-of-the-art strategy developed by Williams et al. [63] as the benchmark strategy. Williams et al. also validated their strategy empirically. The opponents' strategy was obtained from *ANAC Sellers* only. They compare their strategy with two other strategies: a random strategy and Nguyen and Jennings' strategy [47]. Their results show that their strategy outperforms both. Here, we compare CONAN with the second version of their strategy, where all negotiation threads have the same utility (see Sect. 2) and this is due to our assumption that all the sellers provide the same identical resource to be negotiated. We set the simulation parameters based on Williams et al. settings in their empirical evaluation in terms of public deadline, break-off probability and penalties.

**Simulation parameters** The market allows the agents to enter and leave the market at any time. The market simulates the entrance and the exit of the opponents. For each buyer, the market announces the arrival of new sellers and the total number of buyers. The time of the announcement is based on the parameter *Market update time* as shown in Table 4. Table 3 illustrates the variables used in a simulation. The number of new sellers and buyers added will depend on the parameters *Change in market population* and *Change in demand/supply ratio*. For instance, if the change rate in market population is 8 and the demand/supply ratio is 1:1, and the market situation has 5 buyers and 7 sellers, then the market terminates 1 buyer and 3 sellers to reach the correct desired change in the market. These terminated buyers and sellers send an *exit* action to their opponents.

**Table 4** Simulation parameter values

| Variables | Values |
|---|---|
| Buyer's initial price | [300–350] |
| Buyer's reservation price | [500–550] |
| Seller's initial price (agreement ratio wrt to buyer's reservation price) | 10% [750-800]–60% [580–620]–100% [500–550] |
| Seller's initial price (agreement ratio wrt to buyer's reservation price) | 10% [450-500]–60% [380–420]–100% [300–350] |
| Market update time | [2 s, 5 s,10 s] |
| Change in market population | High [30, 40, 50], average [18, 23, 28], low [8, 10, 12] |
| Change in demand/supply | High [10:1, 1:1, 1:10], average [5:1, 1:1, 1:5], low [2:1, 1:1, 1:2] |
| Deadline | Short [30s–90s], average [91s–150s], long [151s–210s] |
| Eagerness | [0.1, 0.3] [0.4, 0.7] [0.8, 1] |

## Performance measurements

To evaluate the performance of CONAN and the state-of-the-art strategy, we use the following performance measurements for a buyer agent.

– *Average utility* [18,47,63]: is the sum of all average utilities of that agent $U_{\text{ag}}(x)$ (see Eq. 1) over the number of negotiation runs $N$:

$$\text{Average utility} = \frac{\sum_{i=1}^{N} U_{\text{ag}}^{i}(x)}{N} \tag{17}$$

– *Percentage of successful negotiations* is the total number of negotiation runs where the agent reaches an agreement with one of the sellers over all runs.

$$\text{Percentage of successful negotiations} = \frac{\text{number of successful runs}}{N} * 100 \tag{18}$$

– *Average number of rounds* [30]: is the average number of negotiation rounds until the accept agreement is reached.

$$\text{Average number of rounds} = \frac{\text{number of rounds per run}}{N} \tag{19}$$

**Experimental set-up** In our evaluation, we allow the *demand/supply ratio* and market population to change during negotiation to create more realistic settings. In addition, as indicated in Table 4, we set the eagerness for all threads to 0.5, the deadline to long $[151s - 210s]$ and the market update time to $10s$. We set the same deadline for all buyers and sellers in the e-market for fair comparison with the benchmark strategy since it assumes public deadlines. (Both buyers and sellers have identical deadlines.) To test that CONAN negotiates using private deadlines, CONAN does not know that buyers and sellers have the same deadline. Also, as we explained in Sect. 4, we set the MRO $= 1$ and MAN $= 100$. We also set the penalty $D = 0.1$ for fair comparison with Williams's et al. [63] strategy.

Similarly, we choose three ranges for the sellers' initial price and reservation price to create three different agreement zones: 10, 60 and 100%. The reason for this choice is to simulate real-life situations, where the buyer does not know the price range of the sellers. CONAN and the benchmark agent run concurrently within the same simulation as competitors, while in the

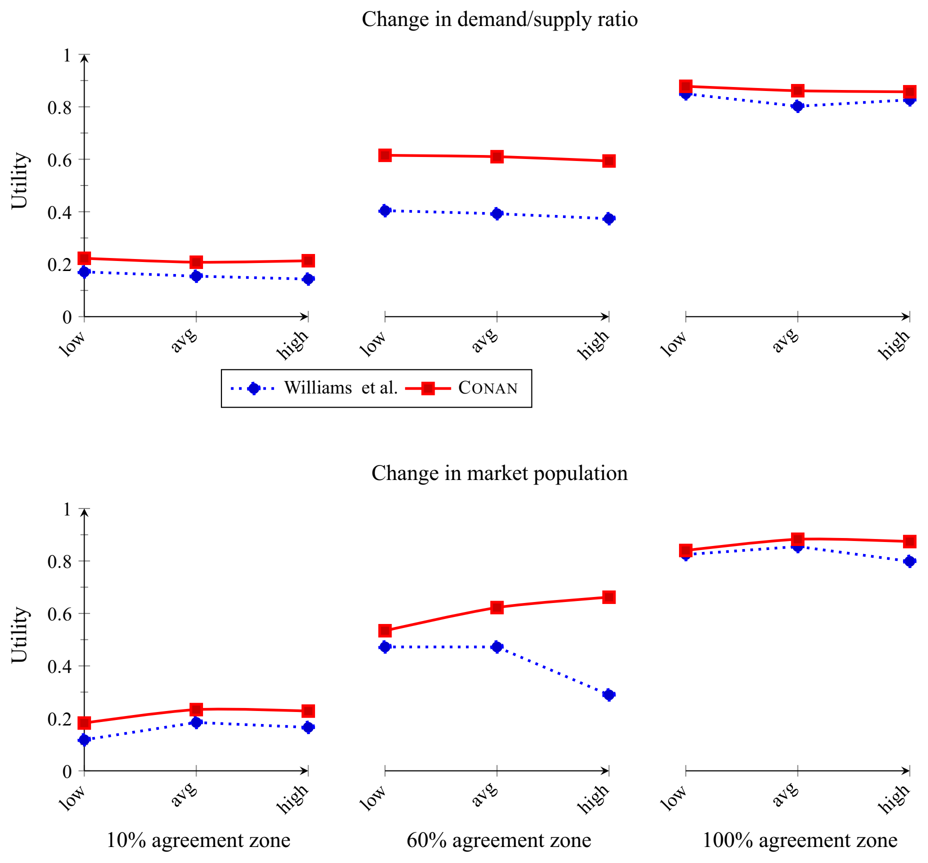


**Fig. 6** Average utility for *Faratin's Sellers*

literature the buyer agents run separately in individual simulations. We run three agents for each strategy (state-of-the-art and CONAN) and run the simulation 100 times for 81 different settings (3 different demand/supply ratio values * 3 different market population values * 3 price ranges * 3 opponent groups) = 8100 runs. We report the average utilities, percentage of successful negotiations and average number of negotiation rounds for those simulations. While some authors have not applied any statistical significance measures [7,33,47], we adopt the standard statistical analysis approaches that has been used in the negotiation literature [63]. We conduct t-tests to report the statistical significance between the utilities of CONAN and the state-of-the-art for each setting and use ANOVA test (analysis of variance) to calculate the statistical significance of the resulting utility between each price range for each group of sellers.

### *5.1.2 Results*

The results for *Faratin's Sellers* are shown in Fig. 6. In the top plot, market population is kept at average and we change the rate for the demand/supply ratio. In the bottom plot, the rate for the demand/supply ratio is kept at average and we change the market population. It can be seen from the plots and from the $p$ value $< 0.05$ that our strategy outperforms the state-

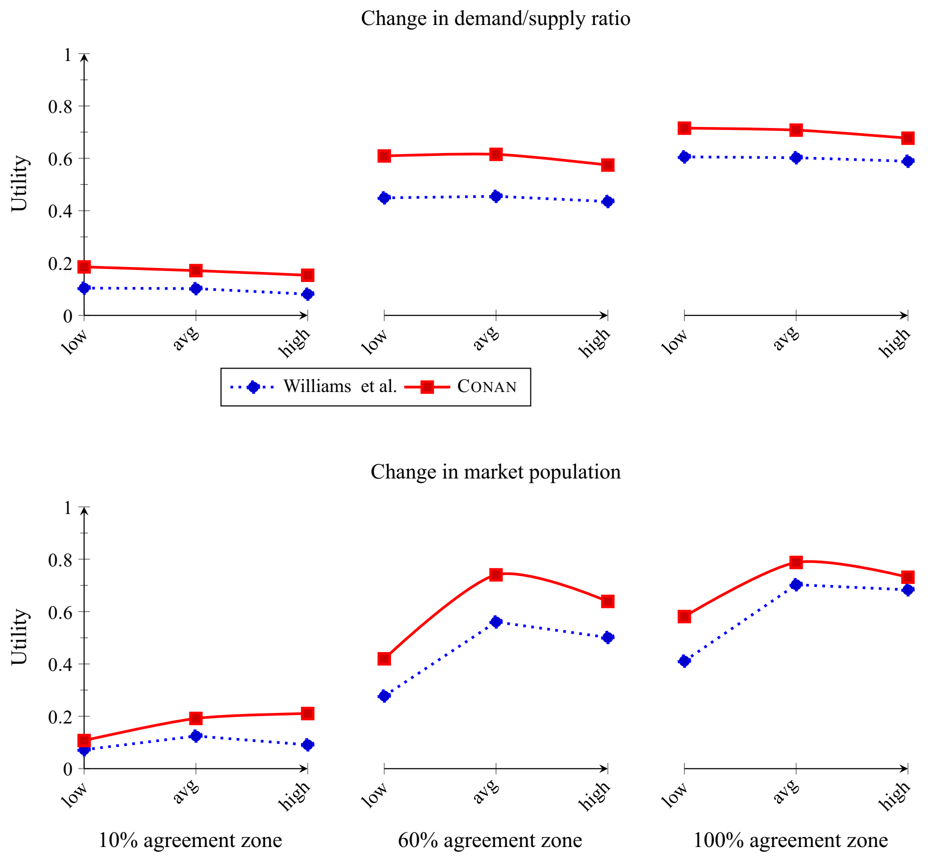


**Fig. 7** Average utility for *ANAC Sellers*

of-the-art according to experimental evidence that is statistically significant. However, the utilities of CONAN and the state-of-the-art are stable during the changes in the demand/supply ratio among the price ranges. On the other hand, the utility of CONAN increases when the market population increases, while the utility of the state-of-the-art decreases when the market population increases. We can observe from Fig. 6 that:

– **CONAN is independent from price range changes** CONAN obtains a significantly higher utility than Williams et al. when negotiating in a high population market and a 60% agreement zone with *Faratin's Sellers*. The most likely explanation for this result is that CONAN takes into consideration that the price ranges between buyers and sellers may differ.

The next result is about negotiating with *ANAC Sellers* and is shown in Fig. 7. Our strategy still outperforms the state-of-the-art ($p < 0.05$). The top plots indicate that each strategy gains the same amount of utility when the rate of demand/supply ratio changes, while in the bottom plots, when the market population is average, CONAN has the highest utility. We can observe from Fig. 7 that:

• **CONAN has high utility in 60% agreement zone:** CONAN obtains significantly higher utility than Williams et al. when negotiating in an average population market and 60%

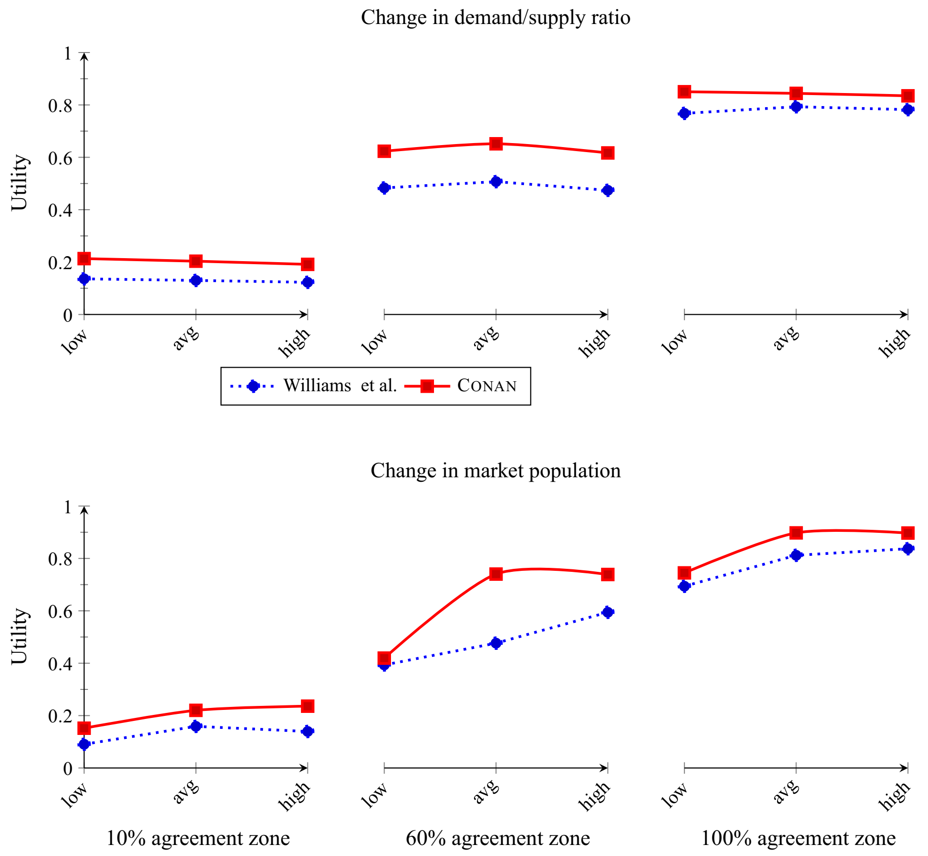


**Fig. 8** Average utility for *All Sellers*

agreement zone with *ANAC Sellers*. We observed in the experiments that *ANAC Sellers* cannot cope with high numbers of agents in the market, so as the number of agents rises, they become slower in producing conceded offers. As a result CONAN accepts offers from the *ANAC Sellers* before the deadline since the utility of the offers $> 0$.

In the third group of sellers, *All Sellers*, the results are shown in Fig. 8. As before, CONAN outperforms statistically significantly the state-of-the-art ($p < 0.05$) within each setting and for each group of settings. CONAN obtains the highest utility when the market population is high. We can observe from Fig. 8 that:

– **CONAN is seller independent** CONAN outperforms the state-of-the-art in all price ranges, demand/supply ratios and market densities when negotiating with *Faratin's Sellers*, *ANAC Sellers* and *All Sellers*;
– **CONAN has high utility in average population** CONAN obtains significantly higher utility than Williams et al. when negotiating in an average population market and 60% agreement zone with *All Sellers*, as explained in *observation 4*.

We can see from Fig. 9 that CONAN gains more successful negotiations than Williams et al. in all the simulation settings when negotiating with *All Sellers*. The top plots indicate that each strategy gains the same percentage of successful runs when the rate of demand/supply

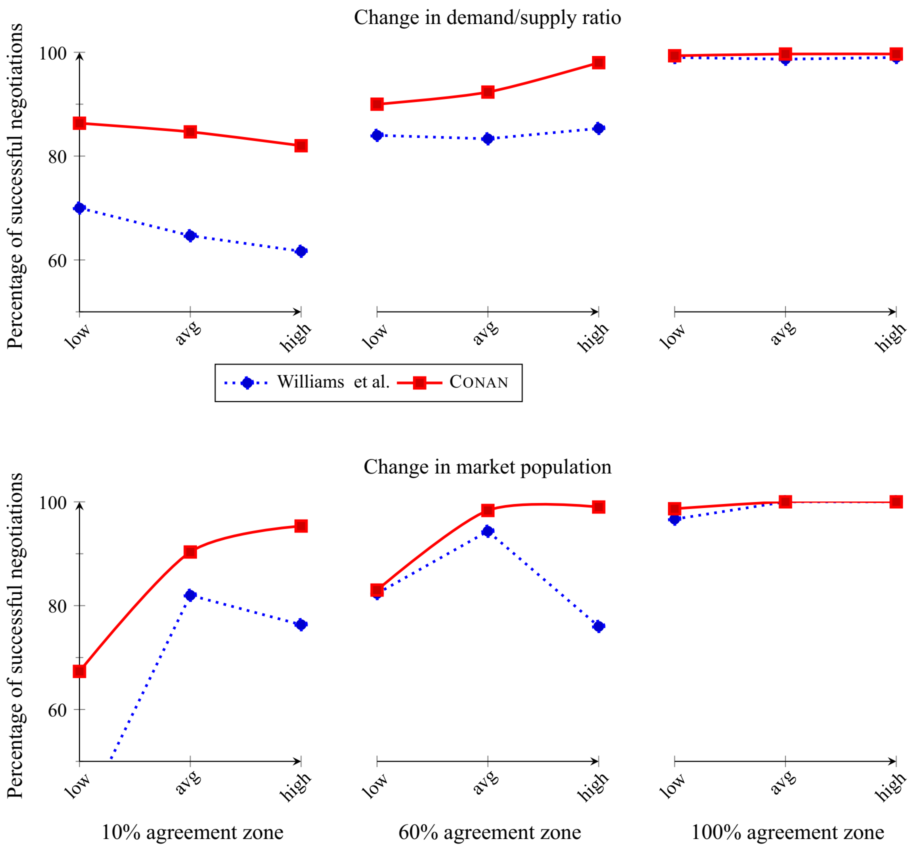


**Fig. 9** Percentage of successful negotiations for *All Sellers*

ratio changes, while in the bottom plots, when the market population is high, Conan has the highest percentage of successful runs. We can observe from Fig. 9 that:

- **CONAN has high successful negotiations** Conan outperforms the state-of-the-art in all price ranges, demand/supply ratios and market densities when negotiating with *All Sellers*, especially in the 10 and 60% agreement zones. This is because it is hard for any agent to get an agreement in the 10 and 60% zones, while it is easier to get an agreement in the 100% zone, since in the 100% agreement zone, the buyer's initial price $\geq$ the seller's reservation price so the utility $> 0$.

We can see from Fig. 10 that Conan conducts less negotiation rounds compared to Williams et al. when negotiating with *All Sellers*. This is because Conan needs more time to think than Williams et al. . Thus, Conan talks less but has more time to think about the best agreement and the best opponent to negotiate with, which minimises the agent's use of communication resources.

The number of rounds is stable (around an average of 100 rounds at low population) for every market population and every demand/supply ratio. On the other hand, Williams et al. have a high number of rounds when the market population is low and a low number of rounds when the e-market population is high.

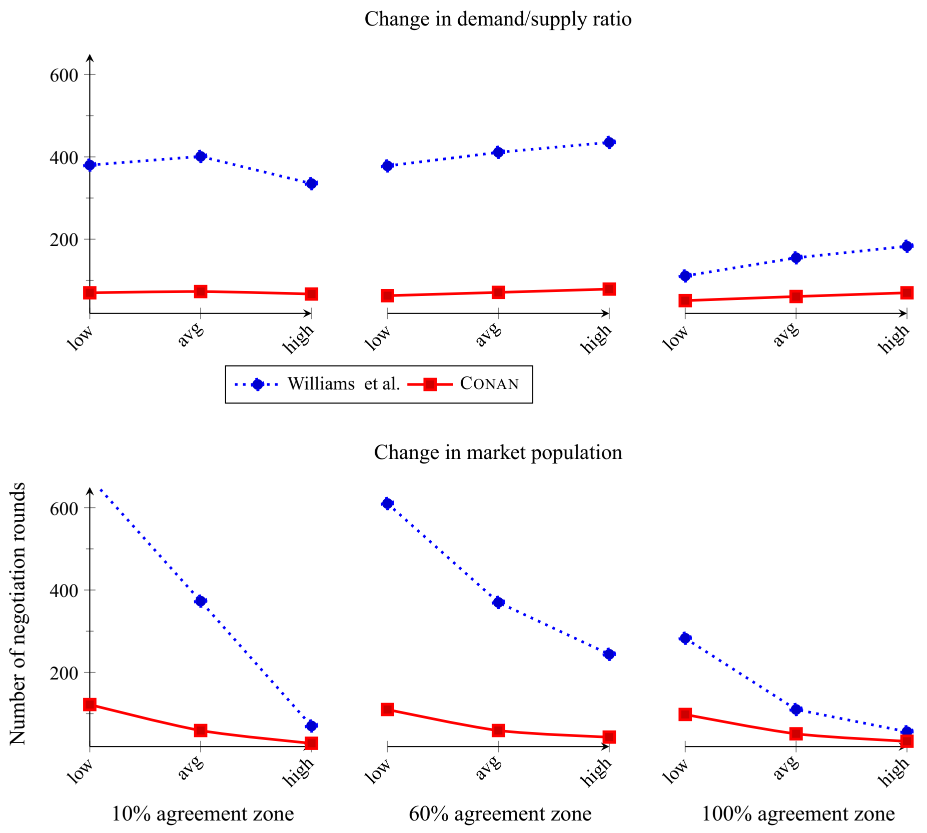


**Fig. 10** Number of negotiation rounds for *All Sellers*

- **CONAN conducts less negotiation rounds** CONAN conducts less negotiation rounds and has a stable number of negotiation rounds than the state-of-the-art in all price ranges, demand/supply ratios and market densities when negotiating with *All Sellers*;

The following is a general observation about the results presented in Figs. 6 7, 8, 9 and 10:

– **CONAN is market-independent** CONAN is a robust and adaptive strategy that is not affected by changes in the market population and/or in the demand/supply ratio in any price ranges. Experimental evaluation of the strategy produces statistically significant results in terms of average utility compared to the state-of-the-art Williams et al. [63]. CONAN also outperforms the state-of-the-art in terms of the percentage of successful negotiations, with fewer rounds. CONAN also achieves the highest utility when the change in demand/supply ratio is low as a result of the market being less competitive, which increases CONAN's opportunity to negotiate with more sellers. Also, in terms of market population, CONAN obtains the highest utility when the market population is high. As a result of the number of sellers increasing, the buyer's opportunity to get an agreement also increases.

## 5.2 Experiment 2: comparing CONAN with random and Faratin's strategies

Given the results of Experiment 1, the aim for this experiment is to determine whether CONAN can outperform other negotiation strategies. In this experiment, we will benchmark more strategies and we will keep the simulation parameters and the experimental set-up as in Experiment 1 (Sect. 5.1). The reason for the second experiment is to make the experimentation more credible and convincing by checking the behaviour of CONAN compared to other, equally famous and competing strategies. We did not combine Experiments 1 and 2 because:

– Experiment 1 considers many performance measurements comparing CONAN with the key benchmark strategy of Williams etal. In Experiment 2 we focus on a general overview of CONAN's performance; thus, it has only one performance measurement.
– Experiment 1 makes sure that CONAN outperforms Williams et al. by running 3 agents for each. In Experiment 2, since we have many buyers to assess, we cannot run more than one agent per strategy in order to keep the low population setting of the e-market as low as possible.
– Experiment 1 tests the performance of Williams et al. and CONAN with different groups of opponents (*Faratin's Sellers*, *ANAC Sellers* and *All Sellers*) since Williams et al. is the key benchmark strategy. In Experiment 2, we need a general overview of CONAN's performance since *All Sellers* represents both *ANAC Sellers* and *Faratin's Sellers*.

### *5.2.1 E-market setting*

As in Experiment 1, we use our negotiation simulator RECON with the following settings:

**Choice of opponents** For our evaluation, we use *All Sellers* strategies as in Experiment 1. The reason for choosing *All Sellers* is that it combines all the opponents that we have, to give a general overview of the performance of CONAN.

**Benchmark strategy** We use the *Random* strategy, and the extended versions of *Faratin's strategies* [18] as the benchmark buyers strategies, since there are no more concurrent strategies that fit to our benchmark selection criteria based on the background review in Sect. 2.

– The *Random* strategy offer generation function is obtained from the GENIUS [3] platform. The offer is randomly generated and the utility of the offer should be $> 0$.
– *Faratin's strategies*: there are eight strategies proposed by Faratin et al. [18]: *Linear*, *Conceder*, *Boulware*, *Resource Dependent*, *Resource Time Dependent*, *Relative Tit-For-Tat*, *Random Absolute Tit-For-Tat* and *Average Tit-For-Tat*.

To make a fair comparison of these existing strategies with CONAN, we extended the Random and Faratin's action selection methods to be able to use the concurrent negotiation actions, which are *reqToReserve* and *cancel*. More precisely, the action selection mechanisms for Random and Faratin's agents have been implemented as follows. The action selection criteria for the offer, accept and exit actions are kept the same as in their original implementations. However, these original implementations are extended with reqToReserve and cancel actions as implemented in the CONAN strategy so that these agents can make fair use of our concurrent negotiation protocol and to enable fair comparison with CONAN.

[3] http://ii.tudelft.nl/genius/.

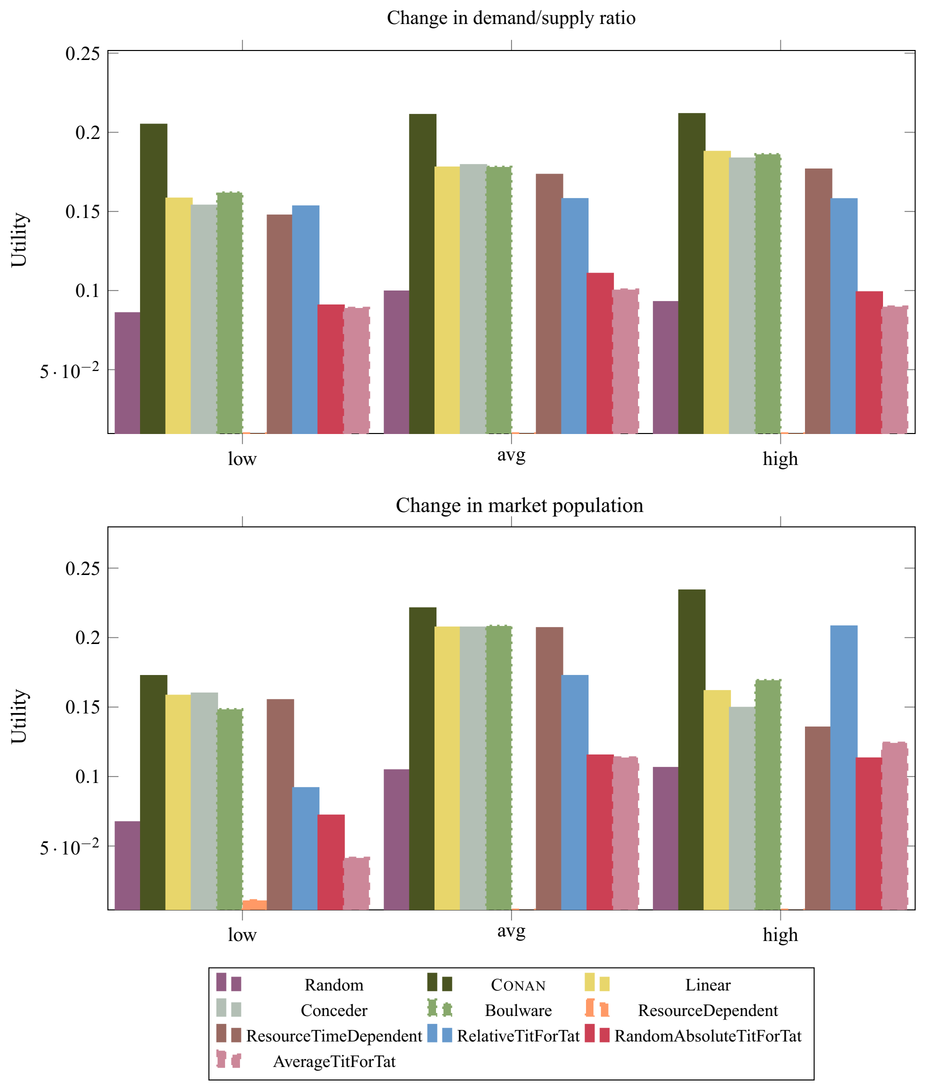


**Fig. 11** Average utility for *All Sellers* with 10% agreement zone

### *5.2.2 Results*

Figure 11 depicts the result of negotiation with *All Sellers* in the 10% agreement zone. As in Experiment 1, in the top plot, market population is kept average and we change the rate for the demand/supply ratio. In the bottom plot, the rate for the demand/supply ratio is kept average and we vary the market population.

It can be seen from the plots and from the $p$ value $< 0.05$ that our strategy CONAN outperforms the Random and Faratin's strategies according to experimental evidence that is statistically significant. From the results we can say that the utility of CONAN, Random and Faratin's strategies is stable during changes in the demand/supply ratio. On the other hand, the utility of CONAN increases when the market population increases, while the utility of *Linear, Conceder, Boulware and Resource Time Dependent* decreases when market population increases.

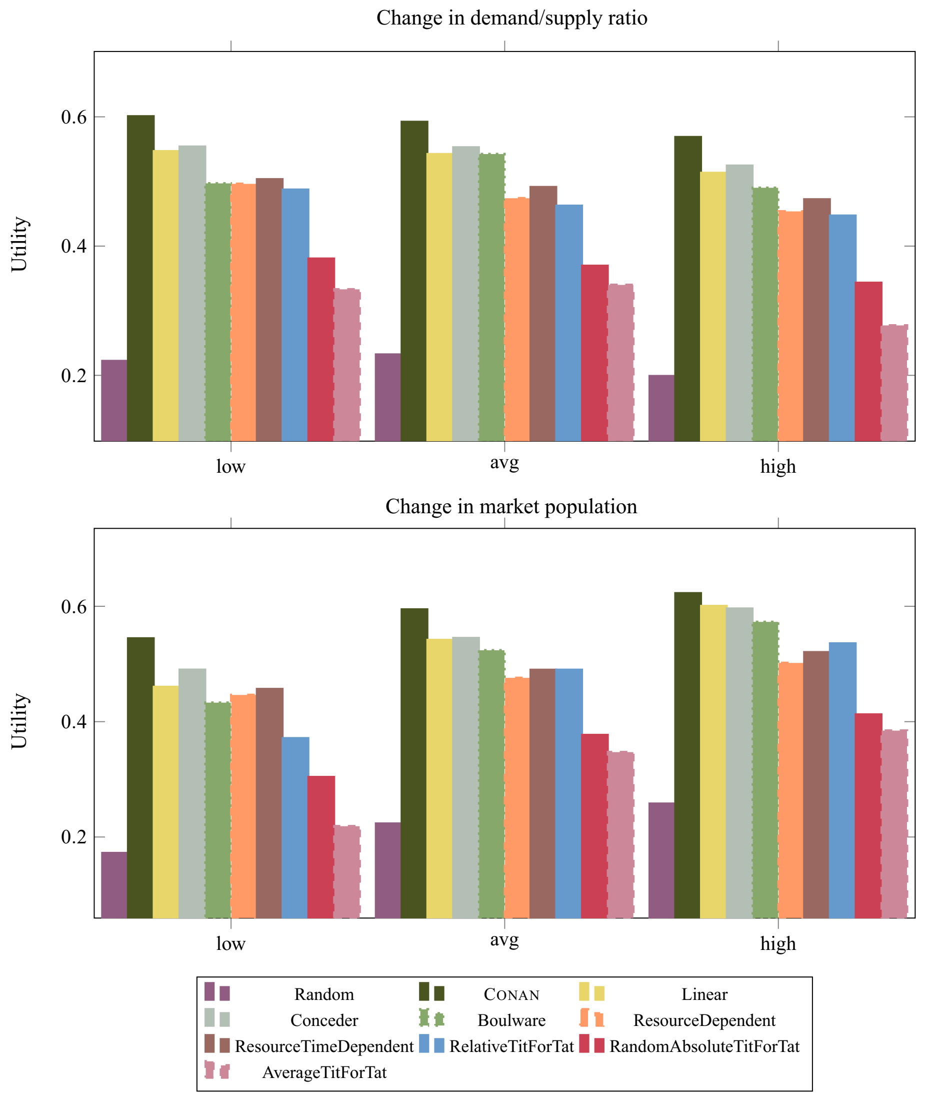


**Fig. 12** Average utility for *All Sellers* with 60% agreement zone

Figure 12 shows the result of negotiation with *All Sellers* in the 60% agreement zone. As in Fig. 11, CONAN still outperforms the benchmark strategies ($p < 0.05$) and the utility of CONAN, Random and Faratin's strategies is stable during changes in the demand/supply ratio. The utility of CONAN increases when the market population increases, and the utility of Random and Faratin's strategies increases when the market population increases.

Figure 13 shows the results for negotiation with *All Sellers* in the 100% agreement zone. As before, CONAN outperforms significantly the benchmark strategies ($p < 0.05$) and the utility of CONAN, Random and Faratin's strategies is stable during changes in the demand/supply ratio. The utility of CONAN increases when the market population increases, and the utility of Random and Faratin's strategies increases when the market population increases.

For this section we combine all the observations from Figs. 11, 12 and 13:

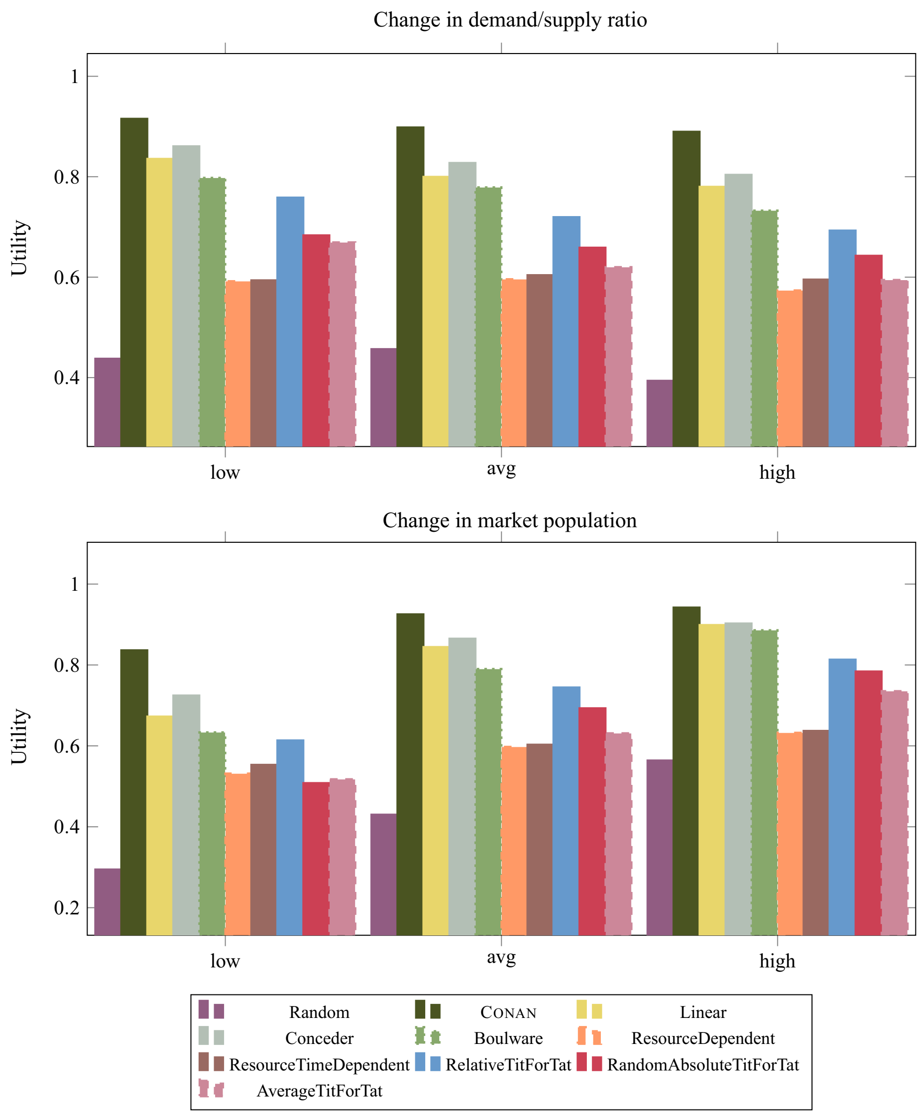


**Fig. 13** Average utility for *All Sellers* with 100% agreement zone

- **CONAN outperforms the Random and Faratin's strategies** CONAN outperforms the Random and Faratin's strategies in the 10, 60 and 100% price ranges, demand/supply ratios and market densities when negotiating with *All Sellers*;
- **CONAN has high utility in 10% agreement zone** CONAN obtains significantly higher utility than the Random and Faratin's strategies (except *Relative Tit-For-Tat*) when negotiating in a high population market in the 10% agreement zone in Fig. 11. As with the Williams et al. strategy, this is because it is easy for any agent to negotiate in the 100% agreement zone and get a better agreement rather than negotiating in the 10% agreement zone, which is more difficult. Also, CONAN obtains significantly higher utility than the

Random and Faratin's strategies when negotiating in a low population market in the 100% agreement zone in Fig. 13. This shows that CONAN can perform well even in a simple market setting where the number of agents is low in a 100% agreement zone.

## 6 Conclusions

We have presented CONAN, a novel heuristic strategy for concurrent negotiating agents for open, dynamic and complex negotiation settings. In this context, we have also discussed how to extend the well-known alternating offers protocol to support concurrent bilateral negotiations and how to use CONAN to generate offers combining environment and self factors. We have also shown how CONAN decides when to accept and reqToReserve offers and when not to further pursue a negotiation thread any more. We have implemented our negotiation framework using the RECON platform [2] to evaluate our work, and we have created realistic e-market experiments by varying parameters such as the market population and the demand/supply ratio. Moreover, we have relaxed most of the assumptions made in previous work, in particular how to deal with private deadlines and the idea of an open market where buyers and sellers may come and go at any time.

We have evaluated the performance of CONAN in various settings and showed that it performs significantly better than the state-of-the-art [63], Random and Faratin's strategies. We have also shown that CONAN's utility is not affected by the changes in the market population, the demand/supply ratio and the price range, including all of them together. Also, our work contributes to the long tradition of practical agent technology in automated negotiation. As our experimental results indicate, users who adopt the CONAN strategy when negotiating (e.g. in e-Bay) will increase their profits compared to other users who adopt the current state-of-the-art strategy.

The extensive experiments that we have performed in this work validate our approach in the selected application domain of e-markets only. Still, the proposed model opens up new possibilities for developing more realistic and flexible models for negotiating agents, especially in areas where the metaphor of e-market [49] has a natural application. We mention here two areas where our contribution is generalisable: (i) service grids [56,60]—where the price of a service by a provider is a single issue under bilateral negotiation with a service client to form contracts, and (ii) cloud computing [5]—where our protocol, buyer strategy and existing seller strategies can be used as a mechanism for allocation of a resource (e.g. a storage) using bilateral negotiations over a single issue (e.g. storage price). Also, the GOLEM agent platform [12] used to build our negotiation testbed RECON [2], has facilities that can include humans as avatars in the experiments (see [14] for an example of user immersed in an e-retailing agent environment), so our approach is generalisable for negotiations between humans and agents [41]. However, a detailed discussion of these issues is beyond the scope of this paper but covered by our plan for future work. This plan includes the following:

- Develop an algorithm to save computation time by determining the most promising sellers as negotiation progresses, and gradually decrease the number of concurrent negotiation threads. This is due to the limited computing resources of the agent, which consume time to reply to all the sellers at the same time.
- Expand our agent model to be able to negotiate over multiple e-markets. Our model only takes into consideration a single e-market. Considering multiple e-markets will facilitate obtaining the item under negotiation faster, and it may increase the agent's utility, since

the agent will negotiate in different e-markets at the same time. In this context, taking into account trust [44] in matching the buyer with different e-markets will also be beneficial.
- Extend our agent model to be able to negotiate over multiple issues. Our current agent only considers one issue (i.e. price) rather than multiple issues (i.e. price, warranty, colour…). As a result of considering multiple issues we will be better positioned to simulate real-life applications, such as negotiating with users. For the simple case of independent issues handled sequentially can be addressed straightforwardly by applying the same approach on each issue—given suitable mapping functions for categorical variables. We will use the item's multiple issues to represent different values of the buyer's eagerness $\mathrm{EG}_b$. For instance, the buyer will have high eagerness to buy a red laptop rather than a blue laptop. Eagerness will help the agent to differentiate between different items and then increase the agent's chance of buying the desired item at a good price.
- Develop a trader agent using CONAN strategy where the agent acts both as a buyer and seller at the same time in a competitive market by expanding the heuristics for deciding actions (Sect. 4.3), in particular the reserve action. We will develop a social welfare metric to measure the agent's utility which combines the buyer's utility and the seller's utility.
- In our experiments so far all participants have been automated software agents. However, in practical applications not all people will choose agents to negotiate on their behalf. As a result, exploring human avatars in GOLEM to negotiate with software agents will be relevant and works such that of Rosenfeld et al. [51] to provide an agent that is able to negotiate with agents and humans at the same time will be key. For this we will need to create a flexible negotiation language for an agent to interact with both artificial and human agents alike. In this context, we will also have to address the challenges of designing decision-making strategies that handle social dependencies to other agents, as in the Color Trails Game of Grosz et al. [26].

**Acknowledgements** We would like to thank the anonymous referees for their valuable comments to a previous version of this paper. We would also like to thank Francesca Toni for valuable comments on a much earlier draft. This paper significantly extends the AAMAS 2014 publication [1].

## References

1. Alrayes B, Kafalı Ö, Stathis K (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, pp 1585–1586
2. Alrayes B, Kafalı Ö, Stathis K (2016) Recon: a robust multi-agent environment for simulating concurrent negotiations. In: Recent advances in agent-based complex automated negotiation, pp 157–174
3. Alrayes B, Stathis K (2014) An agent architecture for concurrent bilateral negotiations. In: Decision support systems III—impact of decision support systems for global environments. Lecture notes in business information processing, vol 184, pp 79–89
4. An B, Gatti N, Lesser V (2009) Extending alternating-offers bargaining in one-to-many and many-to-many settings. In: Proceedings of international joint conference on web intelligence and intelligent agent technology—WI-IAT '09, vol 02, pp 423–426
5. An B, Lesser V, Irwin D, Zink M (2010) Automated negotiation with decommitment for dynamic resource allocation in cloud computing. In: Proceedings of the 9th international conference on autonomous agents

and multiagent systems, AAMAS '10, vol 1. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp 981–988

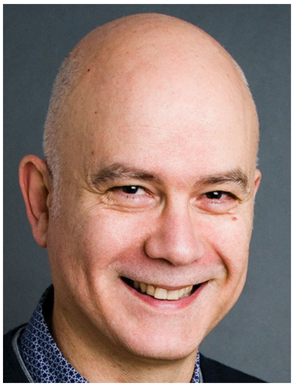
6. An B, Lesser V, Sim KM (2011) Strategic agents for multi-resource negotiation. Auton Agent Multi-Agent Syst 23(1):114–153
7. An B, Sim KM, Tang LG, Li SQ, Cheng DJ (2006) Continuous-time negotiation mechanism for software agents. IEEE Trans Syst Man Cybern B Cybern 36(6):1261–1272
8. Artikis A, Sergot M, Pitt J (2009) Specifying norm-governed computational societies. ACM Trans Comput Logic 10(1):1–42
9. Bakos JY (1991) A strategic analysis of electronic marketplaces. MIS Q 15(3):295–310
10. Bouyssou D, Marchant T, Perny P (2010) Social choice theory and multicriteria decision aiding. In: Bouyssou D, Dubois D, Pirlot M, Prade H (eds) Decision-making process: concepts and methods. ISTE, London, UK, pp 779–810
11. Bouyssou D, Marchant T, Pirlot M, Tsoukiàs A, Vincke P (2006) Evaluation and decision models with multiple criteria: stepping stones for the analyst, vol 86. International Series in Operations Research & Management Science. Springer, Berlin
12. Bromuri S, Stathis K (2008) Situating cognitive agents in GOLEM. In: Weyns D, Brueckner S, Demazeau Y (eds) Engineering environment-mediated multi-agent systems, vol 5049. Lecture notes in computer science. Springer, Berlin, pp 115–134
13. Bromuri S, Urovi V, Morge M, Stathis K, Toni F (2009) A multi-agent system for service discovery, selection and negotiation. In: AAMAS, pp 1395–1396
14. Bromuri S, Urovi V, Stathis S (2009) Game-based e-retailing in GOLEM agent environments. Pervasive Mobile Comput 5(5):623–638
15. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst 25(6):599–616
16. Cohen M, Stathis K (2001) Strategic change stemming from e-commerce: implications of multi-agent systems in the supply chain. Strateg Change 10:139–149
17. Easley D, Kleinberg J (2010) Networks, crowds, and markets: reasoning about a highly connected world. Cambridge University Press, New York
18. Faratin P, Sierra C, Jennings NR (1998) Negotiation decision functions for autonomous agents. Int J Robot Auton Syst 24(3–4):159–182
19. Faratin P, Sierra C, Jennings NR (2002) Using similarity criteria to make issue trade-offs in automated negotiations. Artif Intell 142:205–237
20. Fatima S, Kraus S, Wooldridge M (2014) Principles of automated negotiation. Cambridge University Press, Cambridge
21. Fatima SS, Wooldridge M, Jennings NR (2006) Multi-issue negotiation with deadlines. J Artif Intell Res 27:381–417
22. Fisher R, Ury W, Patton B (1991) Getting to yes: negotiating agreement without giving, 2nd edn. Houghton Mifflin, Boston
23. Forth J, Stathis K, Toni F (2006) Decision making with a KGP agent systems. J Decis Syst 15(2–3):241–266
24. Fujita K, Ito T, Baarslag T, Hindriks K, Jonker C, Kraus S, Lin R (2013) The second automated negotiating agents competition (anac2011). In: Ito T, Zhang M, Robu V, Matsuo T (eds) Complex automated negotiations: theories, models, and software competitions, vol 435. Studies in computational intelligence. Springer, Berlin, pp 183–197
25. García-Lapresta J, Martínez-Panero M, Meneses L (2009) Defining the Borda count in a linguistic decision making context. Inf Sci 179(14):2309–2316
26. Grosz BJ, Kraus S, Talman S, Stossel B, Havlin M (2004) The influence of social dependencies on decision-making: initial investigations with a new game. In: Proceedings of the third international joint conference on autonomous agents and multiagent systems, AAMAS '04, vol 2. IEEE Computer Society, Washington, DC, USA, pp 782–789
27. Hernández J, Mula J, Poler R, Lyons A (2013) Collaborative planning in multi-tier supply chains supported by a negotiation-based mechanism and multi-agent system. Group Decis Negot 23(2):235–269
28. Hindriks K, Jonker CM, Tykhonov D (2009) The benefits of opponent models in negotiation. In: Proceedings of international joint conference on web intelligence and intelligent agent technology, WI-IAT '09, vol 02. IEEE Computer Society, Washington, DC, USA, pp 439–444
29. Hindriks K, Tykhonov D (2008) Opponent modelling in automated multi-issue negotiation using Bayesian learning. In: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems, AAMAS '08, vol 1. IFAAMAS, Richland, SC, pp 331–338

30. Jagabathuni G (2015) Analytical techniques for dynamic negotiation runs in recon. Master's thesis, Department of Computer Science, Royal Holloway, University of London
31. Jennings NR, Faratin P, Lomuscio AR, Parsons S, Sierra C, Wooldridge M (2001) Automated negotiation: prospects, methods and challenges. Int J Group Decis Negot 10(2):199–215
32. Kersten GE, Vahidov R, Gimon D (2013) Concession-making in multi-attribute auctions and multi-bilateral negotiations: theory and experiments. Electron Commer Res Appl 12(3):166–180
33. Kolomvatsos K, Hadjieftymiades S (2014) On the use of particle swarm optimization and kernel density estimator in concurrent negotiations. Inf Sci 262:99–116
34. Kolomvatsos K, Panagidi K, Neokosmidis I, Varoutas D, Hadjiefthymiades S (2016) Automated concurrent negotiations: an artificial bee colony approach. Electron Commer Res Appl 19:56–69
35. Kowalski R, Sergot M (1986) A logic-based calculus of events. New Gener Comput 4:67–95
36. Kraus S (2001) Strategic negotiation in multiagent environments. MIT Press, Cambridge
37. Kurbel K, Loutchko I (2005) A model for multi-lateral negotiations on an agent-based job marketplace. Electron Commer Res Appl 4(3):187–203
38. Lau RY (2007) Towards a web services and intelligent agents-based negotiation system for B2B ecommerce. Electron Commer Res Appl 6(3):260–273
39. Li C, Giampapa JA, Sycara K (2006) Bilateral negotiation decisions with uncertain dynamic outside options. IEEE Trans Syst Man Cybern C 36(1):31–44
40. li Huang S, ren Lin F (2007) The design and evaluation of an intelligent sales agent for online persuasion and negotiation. Electron Commer Res Appl 6(3):285–296
41. Lin R, Krau S (2010) Can automated agents proficiently negotiate with humans? Commun ACM 53(1):78–88
42. Lin R, Kraus S, Baarslag T, Tykhonov D, Hindriks KV, Jonker CM (2014) GENIUS: an integrated environment for supporting the design of generic automated negotiators. Comput Intell 30(1):48–70
43. McGinnis J, Stathis K, Toni F (2009) A formal framework of virtual organisations as agent societies. In: Formal aspects of virtual organisations, pp 1–14
44. De Meo P, Ferrara E, Rosaci D, Sarnè GM (2015) Trust and compactness in social network groups. IEEE Trans Cybern 45(2):205–216
45. Morge M, McGinnis J, Bromuri S, Toni F, Mancarella P, Stathis K (2007) Toward a modular architecture of argumentative agents to compose services. In: Proceedings of EUMAS
46. Nguyen T, Jennings NR (2003) A heuristic model of concurrent bi-lateral negotiations in incomplete information settings. In: Proceedings od IJCAI'03, pp 1467–1469
47. Nguyen T, Jennings NR (2004) Coordinating multiple concurrent negotiations. In: 3rd international conference on autonomous agents and multi-agent systems, pp 1064–1071
48. Nguyen T, Jennings NR (2005) Managing commitments in multiple concurrent negotiations. Int J Electron Commer Res Appl 4:362–376
49. Rahwan I, Kowalczyk R, Pham HH (2002) Intelligent agents for automated one-to-many e-commerce negotiation. In: ACSC '02: proceedings of the twenty-fifth Australasian conference on Computer science, Australian Computer Society, Inc., Darlinghurst, Australia, pp 197–204
50. Rahwan I, Sonenberg L, Jennings NR, McBurney P (2007) STRATUM: a methodology for designing heuristic agent negotiation strategies. Appl Artif Intell 21:489–527
51. Rosenfeld A, Zuckerman I, Segal-Halevi E, Drein O, Kraus S (2016) Negochat-a: a chat-based negotiation agent with bounded rationality. Auton Agent Multi-Agent Syst 30(1):60–81
52. Rosaci D, Sarnè GML (2014) Multi-agent technology and ontologies to support personalization in B2C E-Commerce. Electron Commer Res Appl 13(1):13–23
53. Rosenschein JS, Zlotkin G (1994) Rules of encounter: designing conventions for automated negotiation among computers. MIT Press, Cambridge
54. Rubinstein A (1982) Perfect equilibrium in a bargaining model. Econometrica 50(1):97–109
55. Sim KM (2002) A market-driven model for designing negotiation agents. Comput Intell 18(4):618–637
56. Sim KM, Shi B (2010) Concurrent negotiation and coordination for grid resource coallocation. IEEE Trans Syst Man Cybern B 40:753–766
57. Stathis K, Toni F (2005) The KGP model of agency for decision making in e-negotiation. In: Joint-workshop on decision support systems, experimental economics e-participation
58. Sycara K, Dai T (2010) Agent reasoning in negotiation. In: Kilgour DM, Eden C (eds) Handbook of group decision and negotiation, vol 4. Advances in group decision and negotiation. Springer, Dordrecht, pp 437–451
59. Sycara K, Zeng D (1997) Benefits of learning in negotiation. In: Proceedings of AAAI, pp 36–41
60. Toni F, Grammatikou M, Kafetzoglou S, Lymberopoulos L, Papavassiliou S, Gaertner D, Morge M, Bromuri S, McGinnis J, Stathis K, Curcin V, Ghanem M, Guo L (2008) The ArguGRID platform: an overview. In: Altmann J, Neumann D, Fahringer T (eds) Grid economics and business models, 5th

international workshop, GECON 2008, Las Palmas de Gran Canaria, Spain, vol 5206. Lecture notes in computer science, Springer, pp 217–225
61. West P, Ariely D, Bellman S, Bradlow E, Huber J, Johnson E, Kahn B, Little J, Schkade D (1999) Agents to the rescue? Marketing Lett 10(3):285–300
62. Williams CR (2012) Practical Strategies for Agent-Based Negotiation in Complex Environments. PhD thesis, University of Southampton
63. Williams CR, Robu V, Gerding EH, Jennings NR (2012) Negotiating concurrently with unknown opponents in complex, real-time domains. In: 20th European conference on artificial intelligence, vol 242, pp 834–839
64. Zeng D, Sycara K (1998) Bayesian learning in negotiation. Int J Hum Comput Stud 48(1):125–141

**Bedour Alrayes** is an Assistant Professor in the Department of Information Technology at King Saud University. She received a B.Sc. in Computer Science from King Saud University, Saudi Arabia in 2003, an M.Sc. degree from the same University in 2007 and a Ph.D. in Computer Science from Royal Holloway, University of London in 2016. Her research interests are multi-agent systems, especially automated negotiation, artificial intelligence and decision-making. Her research on CONAN has won second place at the Innovation and Entrepreneurship Prize for Saudi Students in the UK.

**Özgür Kafalı** is a Postdoctoral researcher in the Department of Computer Science at North Carolina State University. His research interests include artificial intelligence, multiagent systems, and computational logic, as well as their applications to e-commerce, e-health, cybersecurity, and privacy problems. Kafalı received a Ph.D. in Computer Engineering from Bogazici University, Turkey.



**Kostas Stathis** is Professor of Artificial Intelligence in the Department of Computer Science at Royal Holloway, University of London and director of the Distributed and Intelligent Computing Environments Lab. He studies the development of autonomous systems with cognitive and social capabilities with the twin goals of understanding human intelligence in symbolic computational terms and to develop applications that support what people do in their everyday activities. He is Fellow of the British Computer Society, a member of AAAI, ACM, IEEE and practitioner of the British Education Academy.