Negotiation strategies in multi-agent systems for meeting scheduling

Rodrigo Rodrigues Pires de Mello, Thiago Ângelo Gelaim, and Ricardo Azambuja Silveira

***Abstract*—An agent not always can single-handed finish a certain task, sometimes he has to interact with others agents to achieve the desired outcome. This interaction can be a time request, in which the agent must spend with the other agent. To address this task, this research focuses on using a multi-agent system for automatic scheduling. During this task, agents may not agree upon the date of a certain meeting or take too much time to decide what action to execute. In order to solve these conflicts, this study applies counterproposal technique and user preferences to implement two different negotiation strategies in a multi-agent system for an automatic meeting scheduler. The scenarios of this work indicate that combining different strategies at certain moments can increase the chance of success during commitment definition, reduce the amount of information shared among agents, avoid privacy issues, raise the expressiveness of agents acceptance region and remove the necessity of a strategy selector.**

***Index Terms*—agent, negotiation, strategy, automatic scheduling.**

I. Introduction

In order to perform an autonomous task in a dynamic environment, the usage of agents can be applied in domains that require too much human intervention, like organizing a trip, scheduling a meeting, and a manufacturing scenario of a factory Stone and Veloso [12]. Sometimes an agent must interact with other agents or humans to be able to achieve their goals. Interacting to achieve some goal is not the only requirement of a multi-agent system. As in other fields of computer science, the amount of time and computational cost to finish a task must be considered during a development of agents Stone and Veloso [12].

The usage of multi-agent systems must consider that sometimes the resources available in a dynamic environment can become limited or unavailable Zhu and Jiang [16]. This kind of situation can create some disputes between agents because they could have different beliefs and goals Kraus [7]. However, agents can use negotiation to solve these differences and achieve a certain state of world Parsons, Sierra, and Jennings [10]. Negotiation can be considered as one of the most important interactions between agents since it is a mechanism that manages interdependencies during execution time Jennings et al. [6]. An agent that negotiate must be able to create proposals, make concessions to others agents, prepare options to solve a problem, and make deals about what must be done Wooldridge [14].

In a scenario where automatic meeting scheduling is performed by agents, negotiation takes an important role during the entire process. In this domain, which is the main focus of this research, we have agents that allocate meetings for a person taking into account their preferences about dates, people, and subjects, in which their beliefs, desires, and intentions are based on a way that they reflect the user's preferences and priorities Zunino and Campo [17].

During a meeting scheduling each agent may have the same main goal, allocate meetings for the person that he represents. Nevertheless, the agents can disagree about some details of this event, like the time, place, and subjects. There are several works that apply users priorities and preferences to be able to solve these conflicts Hossain and Shakshuki [5], Zunino and Campo [17], Lee and Pan [8], Shakshuki et al. [11], and Megasari et al. [9]. Even though these works present their solution for automatic meeting scheduling, most of them do not concern about privacy loss Hossain and Shakshuki [5], Lee and Pan [8], and Megasari et al. [9], where agents can have access to other participants responses about the meeting invitation or apply time or subject preference during meeting allocation Shakshuki et al. [11] and Hossain and Shakshuki [5].

The main contributions of this paper towards the problems of privacy loss and user preferences representation are two negotiation strategies that can be combined to achieve successful meeting allocation. The first one is based on a counterproposal approach, where an agent replies an invite with new possibilities for the meeting. The second strategy is based on the auction, where each agent bids on a certain resource Wurman, Wellman, and Walsh [15], it defines a numeric value that represents the required effort to accept an invite to some meeting. During the negotiation process, the agent that receives a meeting invite will only interact with the agent that proposes the meeting, therefore reducing the information shared in a way that his responses do not contain any information about the user schedule or preferences about time, people, and subject. Another important contribution of this paper is that the strategies presented consume a fixed number of iteration and can be easily incorporated into others scenarios, such as trip organization.

The remainder of this paper is organized as follows: In the section II, we introduce the main approaches used by agents for automatic meeting scheduling. The description of our proposed model of this work is given in the section III. The subsection III-A contains the main features and the necessary information to apply the negotiation strategies during agent's interactions. The subsections III-B and III-C show the approach of creating and responding to a meeting request. When a conflict occurs during a meeting request, the agents engage in a negotiation process and use the strategies defined in subsection III-D, III-E and III-F. The subsection IV presents an empirical study used for test the strategies proposed in this work. The conclusions and future work are given in section V.

II. Related work

Solving meeting schedule problems can be accomplished through Distributed Constraint Optimization (DCOP) techniques Bowring, Tambe, and Yokoo [2], or, as in this paper, with negotiation strategies for multi-agent systems. Despite DCOP consider privacy as a trade-off, privacy loss analysis shows that these algorithms still have this problem Greenstadt, Pearce, and Tambe [4].

Based on the privacy issue that DCOP presents, this section describes the main features extracted from a set of works that was essential to the development of the negotiation strategies for the multi-agent system presented in this paper. Applying user preferences as a factor to decide if a meeting should be accepted plays an important role in automatic meeting schedule Hossain and Shakshuki [5], Zunino and Campo [17], Lee and Pan [8], and Megasari et al. [9]. Hossain and Shakshuki [5], Megasari et al. [9], and Lee and Pan [8] seem to agree that agents could access other agents responses or use a group calendar database during commitment definition. However, Zunino and Campo [17] defend that the agents should release the minimum amount of data of their schedule and preferences to the other participants. The last approach was considered during the implementation of the strategies of this research, where each participating agent interacts only with the agent that proposes the meeting and do not release any information about user schedule or preference.

Shakshuki et al. [11] approach is similar to our research, in which each agent represents a different user or a group, and multiple strategies were applied to meeting allocation. The negotiation strategies are the following:

- First come, first served: when the proposed time was available, the meeting request is accepted;
- Highest rank: this strategy is used when a conflict occurs, the agents make his decision based on user preferences to infer which meeting has a higher priority.
- Voting strategy: when the negotiation process reaches the limit time, this strategy will be applied. Each agent searches for groups of possible times for the meeting and then each one vote for a specific time.

The main difference between Shakshuki et al. [11] and our research is that they do not consider other participants priorities or subject priorities in commitment definition. This approach could lead to cases where a meeting request would be denied if the agent that creates this request does not have a high priority, however, the subject of this meeting or other involved people could be important for the user that is represented by this agent. Based on this approach, our work takes into consideration the subject and other participants of the meeting during the negotiation process.

III. Proposed model

On automatic meeting scheduling, each agent represents a specific person, or a group of people, and communicate with others agents to define the schedule of the meeting Shakshuki et al. [11]. This work assumes that a meeting is an activity that needs at least two people and has a subject to be dealt. Agents have to negotiate to reach an agreement about the time for the meeting.

Definition 1: A meeting can be defined as $M = < A_h, A_i, S, D >$ where:

- A_h is the host agent, the one that will propose a meeting;
- $A_i = \{A_0, A_1, ..., A_n\}$ is a set that contains the agents that will receive a proposal and respond to it;
- S is the main subject of the meeting;
- D is the proposal date of the meeting.

This work assumes that each agent has their own belief base, subsection III-A presents a representation of user preferences and his priorities. To be able to create a meeting proposal and respond to it, the agent has to retrieve the information stored on their belief base, subsections III-B and III-C show the required steps for these tasks, respectively. The counterproposal and effort metric strategies that are used at different moments are presented in subsection III-D and III-E. Another important aspect of this works is the responses analysis that the host agent has to do at the end of every negotiation iteration, the subsection III-F contains this approach.

A. Belief base

In many situations, someone or something has higher priority over others things, some of them may be more important, urgent, or even preferential. In automatic meeting scheduling, every agent has their own belief base with a representation of user preferences and priorities over people, subjects, and time.

Definition 2: A belief base is defined as $B = < P, S, Schedule, P_S >$ where:

- $P = \{< P_1, I >, < P_2, I >, ..., < P_n, I >\}$ is a set that represents an association between a person and his priority towards the user the agent represents, where P_i is a person name and I is an integer varying between 1 and 10;
- $S = \{< S_1, I >, < S_2, I >, < S_n, I >\}$ is a set that associates subjects and priorities, where S_i is a subject that is discussed during a meeting and I is the priority of this matter. When a subject is not known by the agent, a default value is assumed;
- $Schedule = \{ < Day_1, Time_1, I, Status >,$
$< Day_2, Time_2, I, Status >, ...,$
$< Day_n, Time_n, I, Status > \}$
is a set that represents the association between time slots, availability, and priority.
- $P_S = \{< P_1, S_1 >, < P_1, S_2 >, ..., < P_n, S_n >\}$ is a set that contains the list of people the user usually has a meeting with and their association with a subject. It is worth mentioning it that a person can be associated with different subjects, such as P_1.

Every priority is defined by an integer value that varies between 1 and 10, in which 1 represents the maximum priority and 10 the lowest. This is an attempt to recreate the user's preferences about the main features needed to allocate meetings, such as time, involved people, and subject. For example, some people prefer to attend meetings in the morning, therefore the times scheduled in the morning have a low value for time priority Crawford and Veloso [3].

B. *Creating a meeting invite*

When the agent perceives a subject that has not been marked as solved in a meeting, he will try to allocate this subject at the next meeting. The agent's belief base has the necessary information to link the subject with people that have some interest to participate in this meeting.

The next step of meeting invite composition consists on to choose a time to be proposed. As mentioned before, the agent's belief base contains the priority of subjects, times, and people. After the participating members and subject are defined, the agent has to retrieve the subject priority and search, on the user schedule, which times have equal priority and then the first time of this set will be the proposed time for the meeting. The algorithm 1 shows the main steps for creating a meeting invite.

Algorithm 1 Algorithm for creating a meeting proposal. This algorithm returns a message with the host name, the main subject that will be discussed, the proposed time and the involved members.

```
1: function CREATE_MEETING_PROPOSAL
2:     subject_priority=get_most_urgent_unsolved_subject();  ▷ This function returns the set of unsolved subject with the associated priority.
3:     slots=free_slots_prior(subject_priority);  ▷ This function returns the free slots of the most priority subject.
4:     members=find_persons_by_subject(subject_priority);  ▷ This function returns the set of people that are associated with a subject.
5:     return create_message(host, subject_priority, slots, members);
6: end function
```

C. *Responding to a meeting request*

When an agent receives a meeting request, he will access his belief base and find if this meeting request can be accepted. If this time was already scheduled for any other appointment, or the agent that sends a meeting request does not have a high priority, then the agent will use another information before making a decision.

If someone sends a meeting request and this person does not have importance enough to make the agent accept the meeting request, then some other aspects may weight on the final decision. In some cases, the other involved people may have a higher priority, so it is crucial to take this into account before denying the invitation. Another important aspect is that the subject could be an important factor of user preference, therefore the subject can be used at this phase of the negotiation.

Once a meeting request is received and the agent detects the proposed time is not available, this agent will enter into a negotiation process with another agent, which the main goal is to allocate this meeting at another time. The host priority, subject priority, and the people priority play an important role in negotiation execution. In some cases, the agent must use more than one strategy to be able to represent user preferences during negotiation. The subsection III-D describes the first negotiation strategy required to create an answer. Another approach is presented in subsection III-E, in which is defined a strategy that tries to stipulate the necessary effort to attend the meeting. Each strategy presented at subsections III-D and III-E are used in different iterations of the negotiation process.

D. *Creating a counterproposal*

The distance between the current offer and the acceptance region of other agent defines the required time to end a negotiation and make a deal Jennings et al. [6]. To reduce this distance during a meeting schedule, the agent that receives an invite for a meeting can create a counterproposal, when the proposed time cannot be accepted, and presents other possible times for this meeting.

This counterproposal strategy tries to use every information that is inside of a common meeting invite, such as time, participants, and subject. For example, after an agent received an invitation, he will access his belief base and retrieve the host priority of the meeting. If this host has the value 1 as the priority, every available time slot that contains the value 1 as the priority will be chosen to be appended to the counterproposal. To avoid searching through all the slots, this strategy will only consider available slots of the week that the meeting was originally proposed.The algorithm 2 presents every required step to create a counterproposal.

E. *Establishing the required effort to accept a meeting*

Based on auctions protocol, where each participant bids on a certain resource Wurman, Wellman, and Walsh [15] and Bellantuono et al. [1], another strategy was developed to deal with aspects the first strategy does not cover. This strategy is an attempt to create a numeric representation of the required effort to accept a meeting at a certain time.

Definition 3: The effort is defined as: $e = ps + pr + pg + n$ where:

- ps is the subject priority;
- pr is host priority;
- pg is the average of people priorities that have been invited to the meeting;
- h is the proposed time;
- i is the current iteration of the negotiation;
- $H = \{h_1, h_2, ..., h_n\}$ is the set of available slots that have the priorities pr and $pr + i$. The set $D = \{|h_1 - h|, |h_2 - h|, ..., |h_n - h|\}$ represents the module of difference between the available times and the proposed

Algorithm 2 Algorithm for the counterproposal strategy

```
function MEETING_RESPONSE_PRIORITY_BASED(Message proposal)
    if (free_slot(proposal.time()) then
        return generate_response(proposal, true);
    end if
    meeting_priority=priority(proposal.host());   ▷ This function returns the host priority. ▷ This condition assures that a group priority will be taken into consideration if the host priority does not represent a certain importance. The MINIMUM_PRIORITY is a value defined by the agent's beliefs.
    if (meeting_priority>MINIMUM_PRIORITY) then
        group_priority = get_group_priority(proposal);
    end if
    slots=free_slots_prior(meeting_priority);
    if (slots.empty()) then
        slots=free_slots_prior(priority(proposal.subject()));
    end if
    return generate_response_priority_based(proposal, slots);
end function
```

time. Where $min(di) \mid di \in D$ and $di < di'$, therefore, di is the lowest difference between the available time and proposed time.

- n is a normalized value varying between 0 and 10 defined by di.

For example, assuming that an agent receives a message that proposes a meeting at 7:30 a.m on 09/05/2016. After getting value 2 as the host priority and subject priority from his belief base, the agent will access his belief base and search for the slots that have the priority 1 and priority 2, where the second value represents the sum of the host priority and the current iteration of the negotiation. After reaching the end of this search, the agent gets three possible times to allocate the meeting, the retrieved times was on the same day at 7:00, 8:30 and 9:00 a.m. The difference between the times was 0:30, 1:00 and 1:30. Therefore, the time that represents the lowest difference was at 7:00. The algorithm 3 shows the interaction between agent and it's belief base during the effort definition.

The following example shows the main limitation of the strategy mentioned before:

- The host sends a proposal to n agents that he would like to engage in a meeting with;
- Each person defines the required effort to accept the proposal, where higher the value, higher the effort to attend the meeting;
- After the host receives the n answers, there is only information about the time that he proposed, therefore the host agent will have to choose another time and restart the negotiation process. Even if the host agent submits another time of the list that was previously defined during the proposal creation, this could take too much time.

Algorithm 3 Algorithm for effort strategy

```
function MEETING_RESPONSE_
EFFORT_BASED(message)
    if (!free_slot(message.time()) then
        return create_response_effort_based(
message,MAX_EFFORT);   ▷ If the proposed time is not available, a response message with a max effort value is defined. MAX_EFFORT is an integer value defined by the agent's belief.
    end if
    subject_priority=priority(message.subject());
    priority=priority(message.host());
    slots.push(free_slots_priority(priority));
    slots.push(free_slots_priority(priority+i));   ▷ The free_slots_priority returns the available slots of a priority. The i variable represents the current negotiation iteration.
    distance=difference(slots,proposed_time);   ▷ The difference function returns the available time with the lowest difference from the time proposed by the host of the meeting.
    effort=priority+subject_priority+distance;
    return create_response_effort_based(message,effort);
end function
```

To be able to continue the negotiation process of the previous example, the host could retrieve a set of times and append on a message, however, this set could be very large and demand a certain amount of time to be analyzed by the agents that receive this proposal.

Even being simple and limited, this strategy has the purpose of simulating an auction. Its simplicity would not affect others tasks of the agent. It was noticed that this mechanism could be used to decide between two or more times proposed by others agents. After that, the host chooses the time with the lowest effort.

F. Analyzing the meeting responses

When the first iteration of the negotiation ends, the host agent receives the responses of others agents, then he will try to find an intersection between all answers and stipulate a common time for the meeting. If no intersection is detected, a new iteration of negotiation begins and the host agent will scan every response message to find the time that contemplates the largest number of people and the time that maps to the highest priority group. In each iteration, both of the times that was found will be saved and sent again to the participating agents.

At the beginning of the second iteration, each participant will apply the effort strategy, described in subsection III-E, over the two times sent by the host agent. After receiving all the responses from others agents, the host agent will select the time that represents the sum of the lowest effort. After this step, another round of negotiation begins and another message with the selected time is sent to participants agents. This process continues until every agent accept the meeting or the negotiation reach a maximum value of iterations. The figure 1 presents an interaction diagram with every step of the negotiation between agents and the strategies used by them.

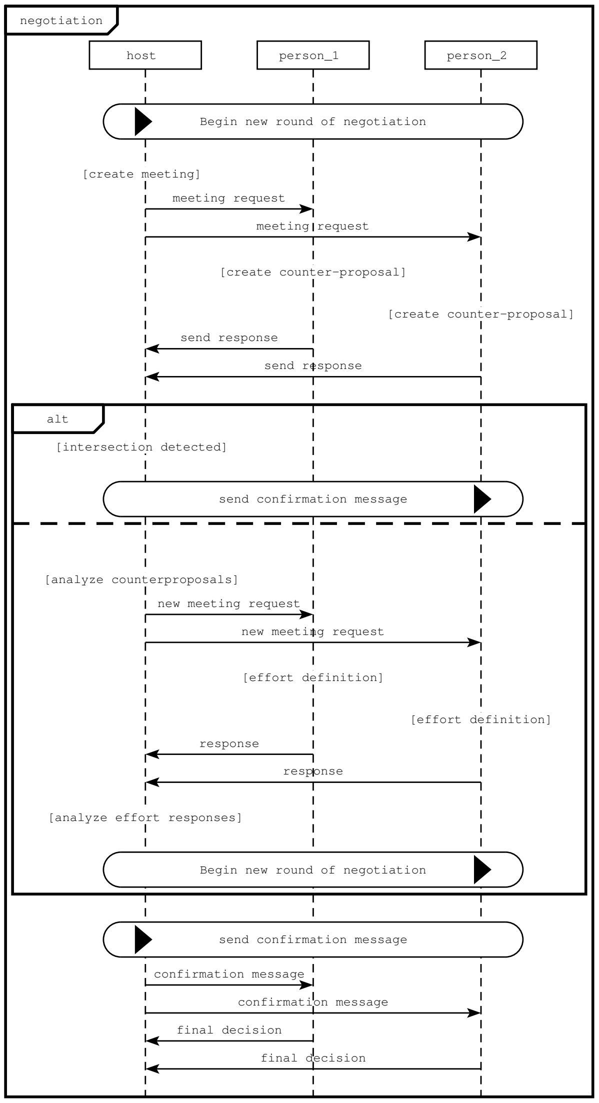

Fig. 1. Interaction diagram of an example of negotiation protocol.

IV. Empirical investigations

In order to simulate the agent's behavior during a meeting schedule. A prototype was implemented to evaluate the proposed negotiation strategies and the belief base that represents the user preferences. The whole process was built as a protocol that agents execute when they are negotiating, however, many aspects were ignored during the development such as speech act and communication languages.

The main tools used during this work was the programming language C++, a Prolog implementation, SWI-Prolog, and an interface between these two languages, that was created by SWI-Prolog as well Wielemaker et al. [13]. The C++ programming language was used for developing the strategies. The interface was used as an access mechanism to the belief base. The agent's belief base was developed using SWI-Prolog, where the priorities and times were represented with facts and data structures, respectively.

TABLE I
Host agent schedule from scenario 1

Time	Priority	Status
7:30	1	available
8:00	1	available
8:30	1	available
9:00	1	available
9:30	2	available
10:00	2	available
10:30	2	available

To make an initial evaluation of the negotiation strategies that was described during this work, three scenarios are presented. The scenario 1 and 2 present simple situations, where the main goal is to explore the expressiveness of each strategy when a conflict occurs. The first scenario focuses on the counterproposal strategy, described in subsection III-D, and exploits the main remarks. The second scenario explores some limitations of the counterproposal strategy and the need for a second strategy that increases the chance of success of the negotiation and the decided time represents a better solution for every participating agent. Scenarios 1 and 2 present a comparison between every strategy, the counterproposal, the effort metric, and our proposal approach that combines both strategies. The third scenario presents important aspects of meetings that occur on daily basis and was adapted from Hossain and Shakshuki [5] work. The main goal of this scenario is to show that group priority can play an essential role to enhance meeting allocation and avoid privacy issues.

It is important to mention that even though the following scenarios uses only three agents, it can be expanded to a different number of agents and messages that will not affect the negotiation process. This is possible because the interaction occurs only with the agent that proposes a meeting and the agent involved in the meeting, therefore other involved agents do not interact with each other. In the following scenarios we use a fixed time for the meeting duration, however it is possible to allocate meeting with different duration.

A. *Scenario 1*

This first scenario shows the interaction between three agents. Agent h1 is the host of the meeting, therefore he will send a meeting request to agents r1 and r2. Both agents r1 and r2 will only interact with h1 during the negotiation process, avoiding the need of sharing private information with others agents. The following tables I and II present the schedule of each agent:

Assuming the proposed subject has the value 1 and it is marked as unresolved by the host agent, the first time slot that will be proposed is at 7:30 a.m. During the negotiation process that uses only the effort strategy, the host agent will propose on the next iteration the times at 8:00, 8:30 and 9:00 a.m and each involved agent will apply the effort strategy for each

TABLE II
R1 AND R2 SCHEDULE FROM SCENARIO 1

Time	Priority	Status
7:30	1	unavailable
8:00	1	unavailable
8:30	1	unavailable
9:00	1	available
9:30	2	available
10:00	2	available
10:30	2	available

time. It is important to remember that the host agent could send all the available times to r1 and r2, however, the effort strategy becomes more flexible during iteration, as mentioned in the subsection III-E.

When the strategy that uses the effort metric is applied in this scenario, the meeting has only one possible time, at 9:00 a.m, to be accepted by r1 and r2. However, it is needed that the times of 7:30, 8:00 and 8:30 be presented as an option, demanding unnecessary interactions between agents. If r1 and r2 create a counterproposal the negotiation can be reduced, where the times with the same priority of the host agent will be proposed, therefore the time at 9:00 will be used when a counterproposal strategy is used. The table III shows the main results of this scenario, highlighting the number of message passing during negotiation:

B. Scenario 2

As mentioned in scenario 1, the counterproposal strategy was superior when it refers to the number of iteration and message passing between agents. Having in mind these characteristics, the second scenario has the main goal to presents the limitation of using only the first strategy. This scenario has the same approach to scenario 1, where h1 interact with r1 and r2 and create a meeting request. The agents schedules is presented at tables IV, V and VI.

The times proposed during the negotiation will start at 8:30 and have 9:00 and 9:30 on the next iterations. Therefore, the number of iteration until both agents accepts the meeting request will be 3.

Using the counterproposal in this scenario has a specific consequence during a meeting allocation. The agents r1 and r2 will search at their belief base and detect a conflict between their schedule and the host agent schedule. After this step, r1 will find that times at 7:30 and 9:00 could be proposed. As agent r1, agent r2 responds to the meeting request with 7:30, 8:00 and 9:00 as alternative times. After agent h1 analyses the messages of r1 and r2, he will found that the meeting could be at 7:30 and 9:00. Then the host agent detects the time at 7:30 as the first intersection, the meeting request will be accepted by others agents, however, the best possible time should be at 9:00 as mentioned on subsection III-E.

The limitation presented before could be avoided if both strategies are applied in this scenario. The counterproposal strategy will define 7:30 and 9:00 as candidates for the time of the meeting. After that, the host agent sends this two times for r1 and r2, and then they could use the effort metric for each time. This approach enables the host agent to choose the time that represents the lowest effort, therefore the time at 9:00. The results of this scenario are presented in table VII.

C. Scenario 3

This scenario was extracted from Hossain and Shakshuki [5] and is used as a comparison with our work. The main focus of this scenario is to show that in some cases it is important to use other aspects of the meeting before rejecting a meeting request. The table VIII shows an interesting case where is possible to notice a well-defined hierarchy among students and professors, and each member of this relation has a certain level of importance.

Just like the previous scenarios, the negotiation process begins when some agent creates a meeting proposal. In this case, professor Darcy invites professor Elhady, and students Mozamma, Wael, and Qianli to a meeting. Table VIII shows that professor Darcy has a high priority towards professor Elhady. However, the same does not occur with the students.

Table VIII shows that professor Darcy does not have a high priority towards students Mozamma, Wael, and Qianli. However, professor Darcy has high priority towards professor Elhady. Hossain and Shakshuki [5] handle this situation in a manner that the agents that represent students wait for the agent of professor Elhady to respond and then follow his decision. This approach handles properly this situation, however, this can be a problem in some situations where users are concerned about their schedule privacy Zunino and Campo [17].

To fully adapt this scenario for our work, we create a hypothetical schedule for professor Elhady and students at tables IX and X. We stipulate that professor Darcy proposes a meeting on 04/09/2017 at 7:30 a.m. As shown in table IX, professor Elhady schedule has a time slot available with the same priority that professor Darcy, therefore he is the only one that will accept the meeting request. The others students schedules do have an available time, however, the priority of this time slot is higher than the priority assigned to professor Darcy. Based on what was presented in subsection III-D, our approach uses the group priority, where the students will analyze which people are involved in this meeting. The group priority will be the average of every participating member, in this case, the value is 2. Table X shows that the group priority 2 can be linked with the time slot proposed by professor Darcy, therefore the agents that represent the students will accept the meeting invitation without using any information about the decision made by professor Elhady's agent.

Even though this scenario seems simple, it represents a normal situation that occurs on a daily basis. Our work defends that the agents should not release any information about user preference or schedule. It is important to mention that our approach does not need any interaction between agents, only the host agent has access to responses, and yet these responses do not contain any confidential information.

TABLE III
RESULTS OF SCENARIO 1.

Strategy	Iteration	Proposed messages	response messages	Total messages
effort	5	8	8	20
counterproposal	2	2	2	8
effort and counterproposal	2	2	2	8

TABLE IV
HOST AGENT SCHEDULE FROM SCENARIO 2

Time	Priority	Status
7:30	2	available
8:00	2	available
8:30	1	available
9:00	1	available
9:30	1	available
10:00	1	available
10:30	2	available

TABLE V
R1 SCHEDULE FROM SCENARIO 2

Time	Priority	Status
7:30	1	unavailable
8:00	1	unavailable
8:30	1	unavailable
9:00	1	available
9:30	2	available
10:00	2	available
10:30	2	available

TABLE VI
R2 SCHEDULE FROM SCENARIO 2

Time	Priority	Status
7:30	1	available
8:00	1	available
8:30	1	unavailable
9:00	1	available
9:30	2	available
10:00	2	available
10:30	2	available

Through scenario 1 and 2, it is possible to notice the main limitations of using only one negotiation strategy during automatic meeting scheduling, even though these scenarios were simple. However, the combination of multiple strategies at certain iterations of negotiation could indicate a better approach for representation of agent's acceptance region, which indicates an improvement in meeting allocation.

Scenario 3 shows the importance that each agent has their user schedule and do not need to interact with other participants, other than the host agent, or share their preferences to be able to allocate meetings. In an indirect way, these interaction reduces the amount of data sent over the network, therefore reducing the required time to process and reason about these messages during negotiation.

V. CONCLUSION AND FUTURE WORK

In automatic meeting scheduling, it is common that each agent represents a different person, therefore they may have different goals, since some people have distinct preferences about the characteristics of the meeting, such as time, for example. To be able to find an intersection between people preferences, agents have to perform a negotiation protocol. In this paper, it was presented two negotiation strategies for automatic meeting scheduling for a multi-agent system.

These negotiation strategies aim to increase the quality of meeting allocation when conflicts occur. The first strategy is based on a counterproposal approach, where each agent uses people, time, and subject priority to propose new times for the meeting. The second strategy tries to define the required effort to attend the meeting that will be held at some specific time.

To be able to test these strategies it was developed a prototype that simulates agent's interaction during the negotiation process. The prototype was developed using C++ and Prolog. The strategies described at subsection III-D, III-E were implemented using C++. The Prolog usage was focused on creating agent's belief base where user preferences are represented. It was used an interface between C++ and Prolog that provides a query mechanism needed by the negotiation strategies.

The empirical investigation presented in subsection IV shows that applying only one negotiation strategy could affect meeting allocation, in which the negotiation could take more iterations or the scheduled time for the meeting is not ideal. Another important result of these scenarios is the agent that receives a meeting request does not need to interact with others involved agents, where the only necessary interaction occurs between the host agent, the one that sends a meeting request, and the agent who will receive a meeting request. This paper shows that an agent does not have to expose the user preferences or priorities to be able to respond to a meeting, where Distributed Constraint Optimization seem to have privacy loss Greenstadt, Pearce, and Tambe [4] and Zunino and Campo [17].

This research defends that negotiation strategies should be combined and this approach could increase the expressiveness of agent's acceptance region representation. Even being an empirical study, the scenarios described in this work could be an indication that using more than one strategy at automatic meeting scheduling increases the quality of user preferences representation during negotiation as mentioned in subsection III-E. Another important feature noticed is the host priority is not the only priority that should weight on the final decision of an agent that received a meeting request, where the subject

TABLE VII
RESULTS OF SCENARIO 2.

Strategy	Iterations	Proposed messages	Response messages	Total messages
effort	4	3	6	13
counterproposal	2	2	2	8
effort and counterproposal	3	4	4	12

TABLE VIII
PRIORITY RELATION OF PROFESSORS AND STUDENTS. ADAPTED FROM HOSSAIN AND SHAKSHUKI [5]. EACH NUMBER REPRESENTS A PRIORITY, WHERE LOWER VALUES REPRESENT HIGH PRIORITIES.

Person	Darcy	Elhady	Mozamma	Wael	Qianli
Darcy	-	1	2	2	3
Elhady	1	-	2	2	3
Mozamma	3	1	-	2	3
Wael	3	1	2	-	2
Qianli	3	1	2	2	-

TABLE IX
PROFESSOR ELHADY SCHEDULE

date	weekday	time slot	priority	status
04/09/2017	Monday	7:30	1	available
04/09/2017	Monday	8:00	1	available
04/09/2017	Monday	8:30	1	available

TABLE X
STUDENTS SCHEDULE

date	weekday	time slot	priority	status
04/09/2017	Monday	7:30	2	available
04/09/2017	Monday	8:00	2	available
04/09/2017	Monday	8:30	2	available
04/09/2017	Monday	9:00	1	available
04/09/2017	Monday	9:30	1	available

or other involved people could be relevant during meeting allocation.

As future work, the presented prototype has to be exposed to more realistic scenarios, in which multiple meetings allocation occurs at the same time and other kinds of commitment or event should be considered. Another important aspect that must be studied is the user preferences and priority update and their role in meeting reallocation.

REFERENCES

[1] Nicola Bellantuono et al. “Multi-attribute auction and negotiation for e-procurement of logistics”. In: *Group Decision and Negotiation* 23.3 (2014), pp. 421–441.

[2] Emma Bowring, Milind Tambe, and Makoto Yokoo. “Multiply-constrained distributed constraint optimization”. In: *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM. 2006, pp. 1413–1420.

[3] Elisabeth Crawford and Manuela Veloso. “Mechanism design for multi-agent meeting scheduling”. In: *Web Intelligence and Agent Systems: An International Journal* 4.2 (2006), pp. 209–220.

[4] Rachel Greenstadt, Jonathan P Pearce, and Milind Tambe. “Analysis of privacy loss in distributed constraint optimization”. In: *AAAI*. Vol. 6. 2006, pp. 647–653.

[5] SM Mozammal Hossain and Elhadi Shakshuki. “A Negotiation Protocol for Meeting Scheduling Agent”. In: *Procedia Computer Science* 21 (2013), pp. 164–173.

[6] Nicholas R Jennings et al. “Automated negotiation: prospects, methods and challenges”. In: *Group Decision and Negotiation* 10.2 (2001), pp. 199–215.

[7] Sarit Kraus. “Negotiation and cooperation in multi-agent environments”. In: *Artificial intelligence* 94.1 (1997), pp. 79–97.

[8] Chang-Shing Lee and Chen-Yu Pan. “An intelligent fuzzy agent for meeting scheduling decision support system”. In: *Fuzzy Sets and Systems* 142.3 (2004), pp. 467–488.

[9] Rani Megasari et al. “Towards host-to-host meeting scheduling negotiation”. In: *International Journal of Advances in Intelligent Informatics* 1.1 (2015), pp. 23–29.

[10] Simon Parsons, Carles Sierra, and Nick Jennings. “Agents that reason and negotiate by arguing”. In: *Journal of Logic and computation* 8.3 (1998), pp. 261–292.

[11] Elhadi Shakshuki et al. “A distributed multi-agent meeting scheduler”. In: *Journal of Computer and System Sciences* 74.2 (2008), pp. 279–296.

[12] Peter Stone and Manuela Veloso. “Multiagent systems: A survey from a machine learning perspective”. In: *Autonomous Robots* 8.3 (2000), pp. 345–383.

[13] Jan Wielemaker et al. “Swi-prolog”. In: *Theory and Practice of Logic Programming* 12.1-2 (2012), pp. 67–96.

[14] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[15] Peter R Wurman, Michael P Wellman, and William E Walsh. “The Michigan Internet AuctionBot: A configurable auction server for human and software agents”. In: *Proceedings of the second international conference on Autonomous agents*. ACM. 1998, pp. 301–308.

[16] Wei Zhu and Zhong-Ping Jiang. “Event-based leader-following consensus of multi-agent systems with input time delay”. In: *IEEE Transactions on Automatic Control* 60.5 (2015), pp. 1362–1367.

[17] Alejandro Zunino and Marcelo Campo. “Chronos: A multi-agent system for distributed automatic meeting scheduling”. In: *Expert Systems with Applications* 36.3 (2009), pp. 7011–7018.