Scheduling meetings through multi-agent negotiations

Jacques Wainer [*]

Institute of Computing, UNICAMP, Brazil

Paulo Roberto Ferreira Junior

Institute of Informatics, UFRGS, Brazil

Everton Rufino Constantino

Institute of Computing, UNICAMP, Brazil

Abstract

This work presents a set of protocols for scheduling a meeting among agents that represent their respective user's interests. Four protocols are discussed: a) the full information protocol if all agents are comfortable in sharing their preference profile and free times , b) the approval protocol if only the preference profile can be shared c) the voting protocol if only the free times can be shared, and d) the suggestion protocol if neither preference or free times can be shared. We use a non-standard metric to evaluate the protocols which aims at maximizing the average preference, but also seeks to reduce the differences in preferences among the agents. The full information and approval protocols are optimal, that is, they achieve the best solution. The voting protocol, in simulation results, achieves the best solution 88% of the time. The suggestion protocol allow for different strategies. Simulation results for different number of agents, different number of solutions, and different strategies are presented. The protocol is shown to be coalition-free. Details of the implemented system is discussed.

Key words: Meeting scheduling, multi-agent, negotiation, calendar

*

Email addresses: `wainer@ic.unicamp.br` (Jacques Wainer), `paulo@wzero.com.br` (Paulo Roberto Ferreira Junior), `constantino.everton@gmail.com` (Everton Rufino Constantino).

1 Meeting Scheduling

Group calendars and meeting schedulers have been one of the first groupware applications [1]. Anyone who has ever tried to schedule a meeting among more than three people can appreciate that some sort of automatic help would be useful. The most common approach implemented so far is the development of a single calendar application that has meeting scheduling facilities. Most commercial systems allow a user to browse the other users's calendars in order to find an empty time slot, and book the meeting in that time slot.

We follow a different approach: we assume that each user uses different calendar systems, according to their own preferences ([2] discusses that indeed people have very different preferences in terms of the form and use of their calendars). The user's calendar is available to an agent that will perform the meeting scheduling for its user. There is already a calendar standard [16] that allows calendar systems to share their calendar data with other applications, in our case, the user's agent. Thus, to schedule a meeting, one selects the participants and instructs one's agent to negotiate a time interval with all the other participants's agents. The problem of scheduling a meeting becomes a negotiation among agents that defend their correspondent participant's interests.

The control of one's own time is a socially delicate issue. Many social measures of status and importance are related to the control of one's and other's time. For example, a more important person can make a less important wait for a meeting, can cancel a meeting with shorter notice, can be more inflexible in scheduling a meeting, and so on. Thus, there are a lot of socially relevant issues related to time management that must be addressed by the designer of a meeting scheduler/group calendar system. We will address in particular the issues of *preferences*, *privacy*, and *homogeneity of the user's satisfaction with the scheduled meeting*.

People have different preferences as to the time to have meetings; they may even have preferences that change according to the kind of meeting: one may prefer organization related meetings in the morning and research meetings late evenings, and so on. The agents should take that into consideration when negotiating a meeting on behalf of its user. With the inclusion of preferences, the problem of scheduling meetings becomes an optimizing instead of a satisfying problem: one has to find a time slot that is good for all participants, instead of just finding one that is free for all of them.

The second issue is privacy. Calendars are usually considered private objects, and their information confidential (but see [3] for the description of an organization where, by default, users not only can browse other people's calendar but also know details about other's meetings, such as topic, participants, and location). There are situations when one does not want one's free time made public: who would hire a consultant or schedule an appointment with a dentist that has a lot of free time? Because of the social attribution of "importance" to people with little free time, one may not be willing to publish one's free time. For symmetry, we will also consider that the participant may not be willing to make his preferences known to the other participants.

The third issue is an experimental one. The standard metric for scheduling a meeting is to maximize the group preference (or satisfaction) with the meeting. But this may leave some people with the feeling that they were forced to accept a low preference time for the "greater good." If strong, such feeling could disrupt the creation of a good group environment for the meeting. So, we propose a metric that aims at homogenizing the satisfaction of all the participants, *as well* as the group satisfaction. Thus, each participants can have some assurance that the others have the similar preference for the meeting as he does.

Multi-agent meeting scheduling is an active research area. Some of the work in this area is discussed in section 5, but in general research in this area falls into two main categories:

- research on agents that learn its user's preferences, [4–8] among others. This line of research, which we will call *adaptive* agents, usually is more interested in the adaptability of the agent to its own user – the negotiation and privacy issues are not deeply dealt with. For example, [5] describes such an adaptive agent which uses a protocol in which all agents share their free times.
- research on the multi-agent aspects of the negotiation, [9–12], among others. This line of research is more concerned with defining a single negotiation protocol, in most cases a protocol that allows for many simultaneous meeting negotiations, and study the efficiency of the protocol, that is, how long does it take to achieve a solution. For example [12] solves the meeting scheduling problem as a distributed dynamic constraint satisfaction problem. The problem is dynamic because the agents are engaged in simultaneous negotiations and thus their constraints change during the negotiation. Furthermore, the paper defines two

levels of constraints - *hard* constraints refer to the availability or unavailability of times intervals to schedule the meeting for each agent, and *soft* constraints refer to the agent's preferences regarding the different time intervals. The soft constraint can be relaxed during the negotiation if no solution was found.

This paper falls more into the multi-agent negotiation research line but different than most research in this line:

- we propose four protocols, one for each level of privacy required,
- we do not allow for simultaneous meeting scheduling.
- we are more interested in the efficacy of the protocols (the quality of the schedule) than the efficiency (the time or number of messages needed to achieve the solution).
- we use a nonstandard metric to evaluate the quality of a schedule, which attempts to balance the average preference for the schedule with an homogeneity of the preferences.

We assume that the agents are fully automated, and thus the negotiation is performed without the participant's intervention. In experiments with the implemented system negotiations never took more than a few seconds. That is why we do not allow for simultaneous meeting negotiations - it is very unlikely that an agent will be called to negotiate a new meeting while a previous negotiation is taking place. If the user has to intervene during the negotiation, for example as it is expected in the system described in [5], then the number of messages, or rounds is an issue - one would not like to bother the user with many interventions in the negotiation. But we assume that the agents will negotiate automatically and autonomously, and thus the efficiency in terms of number of rounds in the negotiations is less of concern.

This paper is organized as follows. Section 2 describes the formal definitions and the metric for evaluating the resulting schedule. Section 3 discusses the different requirements regarding privacy, and presents four different protocols to solve the meeting scheduling problem. Section 4 discusses the simulation results for some of the protocols - the ones that are not optimal. Section **??** discusses the implemented system. Finally section 6, discusses the related work in more details and the limitations of the work.

This paper is an extension of a previous work [13] by two of the authors. The protocols were first defined in that work, but the metric of evaluation used was different. Some of the strategic variations for the suggestion protocol are also not present in that work.

2 Formal definitions

For the scheduling purpose, a meeting m is a 6-tuple $\langle A, \bar{a}, h, l, w_b, w_e \rangle$ where:

- $A = \{a_1, a_2, ..., a_n\}$ is the set of agents invited to the meeting.
- $\bar{a} \in A$ is the owner of the meeting, that is, the agent that called the meeting.
- h is the host that conducts the negotiations for scheduling the meeting. It may be the case that $h \in A$, if one of the members of the meeting takes into its own hand the task of scheduling the meeting (in this case usually $h = \bar{a}$), or $h \notin A$ if the moderator of the scheduling is not one of the participants (for privacy reasons).

- l is how long the meeting is planed to last
- w_b and w_e are the beginning and the end of the window for scheduling the meeting. We assume that meetings must be scheduled within an interval - a meeting to prepare a presentation must be schedule before the presentation itself, and the meeting to evaluate the presentation must be scheduled after the presentation. w_b and w_e are the limits of such a window for scheduling the meeting.

Each agent a_i has a calendar that maps each **time slot** to either a previously scheduled meeting, or to no meeting. In the latter case we will say that the time slot is **free**. For our purposes, the calendar of an agent a_i is represented by the set $\text{Free}(a_i)$ of free time slots. That is, if $ts \in \text{Free}(a_i)$, then ts is a free time slot for agent a_i.

We define a **time interval** as a sequence of one or more adjacent time slots. We will say that a time interval t is free, that is, $t \in \text{Free}(a_i)$ if, for all time slots ts in t, $ts \in \text{Free}(a_i)$. The duration of a time interval is the number of adjacent time slots in it. When there is no risk of ambiguity we will refer to both free time slots and free time intervals as **free time**.

Each agent a_i has a **preference function** $W_i(m, t)$. The higher the value of $W_i(m, t)$, the more "preferred" is for agent a_i to have the meeting m scheduled at time t.

For a given meeting $m = \langle A, \bar{a}, h, l, w_b, w_e \rangle$, we define $\text{Poss}(m) = \{t_1, t_2 \ldots t_k\}$ as the set of all possible time intervals to schedule the meeting. That is, for all $t_j \in \text{Poss}(m)$:

- $t_j \in \text{Free}(a_i)$ for all agents $a_i \in A$,

- duration of each t_j is equal to l,
- each t_j starts after or at w_b and ends before or at w_e.

2.1 Metric of evaluation

Agent a_i **preference** for scheduling the meeting m at the time interval $t \in \text{Poss}(m)$ is:

$$\alpha_i(t) = \frac{1}{|t|} \sum_{ts \in t} W_i(m, ts).$$

that is, the average user preference over all time slots that make the time interval.

If there are N participants in the group, the **group utility** with scheduling meeting m at the interval t is:

$$\beta(t) = \frac{1}{N} \sum_i \alpha_i(t) - \sqrt{\frac{\sum_i (\alpha_i(t) - \frac{1}{N} \sum_i \alpha_i(t))^2}{N-1}}$$

The definition above is the mean of the users' preferences ($\frac{1}{N} \sum_i \alpha_i(t)$) minus the standard deviation of the preferences ($\sqrt{\frac{\sum_i (\ldots)^2}{N-1}}$). Thus, the metric tries to maximize the average of the users preferences but penalizes if the dispersion of the preferences is too large (because that increases the standard deviation).

The **best schedule** for the meeting m, denoted as m^+ is the time interval t that maximizes $\beta(t)$. And the **worst schedule** for the meeting (m^-) is the time interval that minimizes $\beta(t)$.

The **optimization degree** of scheduling the meeting at the time interval t is:

$$N(t) = \frac{\beta(t) - \beta(m^-)}{\beta(m^+) - \beta(m^-)}$$

Besides the degree of optimization, we will also use the **group rank** of the time interval t, denoted as $R(t)$. Given the set of time intervals $t_i \in \text{Poss}(m)$ lets order them in the sequence $\langle t_1, t_2, \ldots t_i \ldots t_n \rangle$ if $\beta(t_i) > \beta(t_{i+i})$ or if $\beta(t_i) = \beta(t_{i+i})$ and t_i is before t_{i+1}. We will define the rank of each time interval as: $R(t_1) = 1, R(t_2) = 2, \ldots, R(t_i) = i$.

In order to measure how satisfied a user is with scheduling the meeting at a particular time interval t, we will define the **adjusted user satisfaction** as:

$$\frac{\alpha_i(t)}{\max_{x \in \text{Poss}(m)} \alpha_i(x)}$$

that is, the relation between the user's preference for the scheduled time interval and the user maximal preference among the possible intervals to schedule the meeting.

3 The negotiation protocols

3.1 Levels of privacy

As we mentioned above, we consider that a participant may not want to share either his free time information or his preferences with the other participants.

In our protocols, each agent communicate only with the host, but the logs of this communication may or may not become public to the other agents. In another dimension of privacy, the participant may not want the logs of his messages with the host to be made public, since they may reveal too much of the agent's preferences and behavior.

Finally, the third dimension of privacy is whether the participant trusts or not the other agents in the scheduling to be the host that manages the negotiation. If the participant does not wish the other participants to know about his negotiation behavior, then the host cannot be one of the agents in the negotiation. Thus, the participant may request that the host that will conduct the negotiation should be an outside entity, a public domain, trusted meeting scheduler host, somewhere in the Internet.

The first dimension **information content**, defines what information will be made available during the negotiation. It defines four modes of interaction:

full information the user has no problem in making both free time and preferences information available.

free time the user can make his free time information available, but not his preferences.

preference the user can make his preferences available, but not his free time.

no information the user does not wish to make either the preference or the free time information available.

The second dimension, which we call **scope**, defines two modes of interaction:

public all information between the agent and the host can be made available to the other agents.

private the information exchanged between the agent and the host cannot be made available to the other agents.

And the third dimension, which we call **location**, defines two modes of interaction:

internal where the host can be the agent of one of the participants (usually the one calling the meeting).

external where the host cannot be one of the other agents.

As we mentioned, in this paper we are more interested in the efficacy of the protocols, than in their efficiency. Regarding efficacy, two important properties of the protocols are whether they are **optimal**, or, at least, **complete**. A protocol is optimal if it always computes the best possible schedule for the meeting (or fails if there is no possible schedule). A protocol is complete if it will not falsely determine that a meeting is not possible when it really is. Of course, an optimal protocol is also complete.

We will discuss in details efficiency in this paper, except for the upper bound to the number of rounds and number of messages needed to achieve a solution. We will consider a *round* an exchange from all agents to the host, followed by the host's answer, or a message from the host to all agents, followed by the agents' responses. We will use *half-round* to refer to an one-way flow of messages. We will assume that each message carries one *piece of information.* The protocols below will exchange free time intervals, agreement or disagreement with a proposed schedule, preferences, and so on. We will call each of these a piece of information and each message will carry one of such piece of information. Finally, we will use N for the number of participants and M for the number of time intervals in the scheduling window.

3.2 Negotiation parameters

To schedule a meeting the agent calling the meeting ($\bar{a}$), sends to all other agents in A, an **invitation for a meeting**, a tuple $\langle A, \bar{a}, l, w_b, w_e \rangle$, where all symbols have the same meaning as in the meeting definition.

Each agent in A sends to $\bar{a}$ a triplet $\langle$ information-content, scope, location $\rangle$, which represents the lowest level of privacy it will accept for the negotiations to schedule this meeting. For each of the three items above, $\bar{a}$ computes the supremum of all the agents' choices according to the lattices in figure 1.

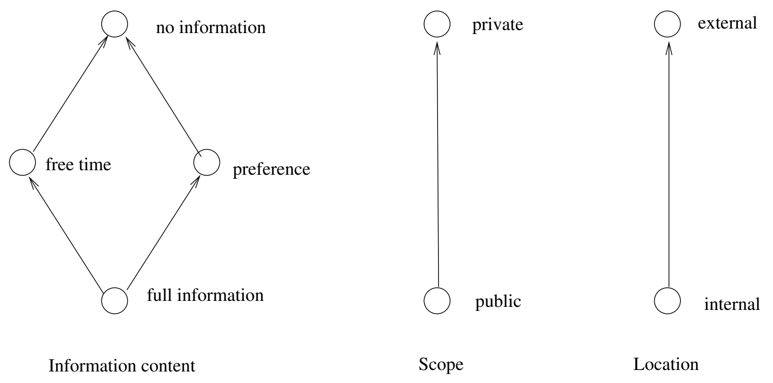

Fig. 1. Lattices of the three dimensions of privacy

If the group decides on an external location, $\bar{a}$ asks each agent to suggest a set of external hosts. $\bar{a}$ computes the union of these suggestions, and resubmits it to the agents. Each agent answers with the subset it accepts as the hosts for the negotiation. The $\bar{a}$ computes the intersection of all lists, and if it is not empty, it will query each of the hosts in the intersection to check if it is active. The first host which is active is selected as the host and its identification is returned to the participants. If the scope is public, then the $\bar{a}$ informs the host to make it available the messages from each agent so they can be retrieved by other agents. The negotiation of the meeting can now start.

3.3 Full information protocol

The **full information** protocol is used when none of the participants have concerns about sharing, at least with the host, both their free time and preference information. The protocol is very simple. Each agent sends to the host their free time within the scheduling window, and the corresponding preferences. The host computes the set $\mathrm{Poss}(m)$ as the intersection of the free intervals for all participant, and using the preference information computes the best schedule m^+. If the set $\mathrm{Poss}(m)$ is empty, the host informs each agent that the meeting cannot be scheduled. Otherwise the host returns m^+ to each agent.

The full information protocol is optimal. The host has the correct information regarding the free times and preferences for all agents, and it computes the time interval with maximal group utility. That is the best possible schedule for the meeting.

The full information protocol achieves a solution after one round. The amount of information exchanged in the round is at most $N * 2 * M + 1$. Each of the N agents send at most $2 * M$ pieces of information or messages: if an agent has no previously scheduled meeting, it will have M free intervals and M corresponding preferences for them.

3.4 Voting protocol

The **voting** protocol is used for the free-time mode. Each agent sends to the host its free intervals within the scheduling window. The host returns to each agent the set Poss(m). If the set is empty, the host informs them that the meeting cannot be scheduled. Otherwise, each agent ranks the set of intervals received from the host in decreasing order of preference. The ranking allows for grouping, so an agent can rank t_4, t_6, and t_7 as the most preferred set of intervals, and t_5 and t_2 as the second preferred set, and so on.

The host then approximates a preference function for each user, from the rank of the sets: all time intervals in the most preferred set receive the preference of 10, and the intervals in the least preferred subset receive the preference 1, and linearly between these two extremes, for the other ranks. For example if a particular agent ranks $\{t_4, t_6, t_7\}$ first, ranks $\{t_5, t_2\}$ second, ranks $\{t_3\}$ third, and ranks $\{t_1, t_8\}$ last, then for that user, the host defines a preference function such that the first set receives preference 10, the second receives 6, the third receives 3, and the last set receives preference 1.

Using the approximation to each user's preference function, the host then computes the interval with the highest group utility, which is returned as the scheduled time for the meeting.

The voting protocol is not optimal, but it is complete. Section 4.1 will discuss the efficacy of the protocol. The protocol achieves a solution in two rounds, and the number of messages exchanged is at most $N*M+M+N*M+1$, where each term corresponds to each half-round.

3.5 Approval protocol

The **approval** protocol is used in the preference mode. Each participant sends to the host its preference profile, that is, the preference for all time intervals between w_b and w_e, even the one that are not free.

The host assumes that every agent has all time intervals between w_b and w_e free, computes the group utility for each time interval, and order them by decreasing group utility. The host then sends one time interval at a time to the agents, which accept it or not. If at least one agent does not accept the time interval, the host submit the next best time interval to the agents' approval. When all agents accept a time interval, it is chosen as the time for the meeting.

The approval protocol is optimal. The proof is by contradiction; let us suppose that the host creates the ordered list of time intervals $\langle t_1, t_2, \ldots t_x \ldots \rangle$ by decreasing group utility, and that t_x is the best schedule for the meeting. That means that for $i < x$ at least one agent does not have t_i as a free time, and all have t_x free. The host will present t_1, t_2, and so on, in that order, as possible schedules for the meeting, and at least one agent will not accept t_i since it is not free for it. t_x will be the first interval accepted by all (if they do not lie - more on this later), and thus the protocol is optimal.

The protocol achieves a solution in at most $1+M$ rounds - the first half-round for the agents to send the preference profile, and at most M rounds in which the host sent a possible schedule and the agents approve it or not, plus a half-round for the host to inform the solution. The total amount of information exchanged is at most $N * M + (1 + N) * M + 1$, where the first term corresponds to the first half-round, the second to the proposal/approve/disapprove rounds and the last term is the last half-round.

Lying is a strategic behavior all agents can follow. In the typical lying scenario an agent will say that time intervals with a preference below a certain threshold are not free and thus the meeting, if scheduled at all, will not be scheduled at these low preference intervals. Lying is possible in all protocols including the full information, the voting protocols discussed above, and the suggestion protocol discussed below, and will achieve the same effect of increasing the user satisfaction with the schedule at the expense of reducing the number of possible time intervals to schedule the meeting. In this paper we will not discuss the effects of lying in the protocols.

3.6 Suggestion protocol

The **suggestion** protocol is used in the no information scenario - the users do not want to divulge either their preferences or free time information. Intuitively, the protocol is based on the idea of a *suggestion*. In each round of the negotiation, each agent must send a *new* suggestion for the meeting. A suggestion is a time interval that the agent will accept as the time for the meeting. The host groups the suggestions, remove the information of which agent made which suggestion, and send them back to all agents. Each agent can see which other suggestions were proposed in the previous round and choose its own next suggestion taking that into account.

Both host and agents retain the state of the negotiation, so the host has only to return the new suggestions proposed in the previous round. Similarly, in each round, the agent only need to send a new suggestion to the host. If the agent has no new suggestions, it sends a particular message ($\perp$) to indicate that.

Algorithm 1 describes the protocol as an algorithm. S^j is the cumulative set of suggestions made until the beginning of round j. R_i^j is the set of suggestions made, until round j by agent i. r_i^j is the suggestion made in round j by agent i. At each round, given the new suggestions r_i^j, the host updates the lists R_i^j and S^j. Time intervals that were added in the round are returned to the agents as the new suggestions. At the end of a round, if an interval in the list has been proposed by all agents, it is set as the time for the meeting. If more than one interval have been proposed by all agents, the host selects the earliest interval for the meeting. Finally, if in a round, all agents send the $\perp$ message, the meeting cannot be scheduled and the agents are informed.

Algorithm 1 The suggestion protocol

```
the host sets $S \leftarrow S^0 \leftarrow \emptyset$
the host sets $R_i^0 \leftarrow \emptyset$
$j \leftarrow 1$
**loop**
  the host sends to all agents the set $S^j - S^{j-1}$ of all new suggestions made at round $j$
  **for all** agents $a_i$ **do**
    **if** there is a new time interval $r_i^j \in \text{Free}(a_i)$ and $r_i^j \neq r_i^k$ for $k < j$ **then**
      $a_i$ answers $r_i^j$
    **else**
      $a_i$ answers $\perp$
    **end if**
  **end for**
  the host sets $R_i^j \leftarrow R_i^{j-1} \cup \{r_i^j\}$
  **if** $\cap_i R_i^j \neq \emptyset$ **then**
    **return** the earliest of all $\hat{x} \in \cap_i R_i^j$ as the schedule for the meeting.
  **else if** for all agents $i$, $r_i^j = \perp$, and $\cap_i R_i^j = \emptyset$ **then**
    **return** the meeting cannot be scheduled
  **end if**
  the host sets $S^{j+1} \leftarrow \cup_i R_i^j$
  $j \leftarrow j + 1$
**end loop**
```

3.7 Analysis of the suggestion protocol

The suggestion protocol is complete, but not optimal. The proof of completeness is as follows. If it is possible to schedule the meeting at t_x, and since each agent must send at least a new suggestion at each round, then after at most M rounds, all agents will have suggested t_x stopping the protocol.

Regarding efficiency, the maximum number of rounds is M, and the total number of messages is $M * (N + N)$, where in each round there are at most N new suggestions from the agents to the host, and the corresponding answer from the host with the N new suggestions.

3.7.1 Strategic Alternatives

The suggestion protocol allows for strategic variations regarding the agents behavior. An **egotist** agent will try to schedule the meeting at its own best times, at the expense of providing some information about itself to the group (specially if the negotiations are public). The egotistic strategy is to rank its set of free intervals in decreasing order of preference $\langle x_1, x_2 \ldots x_k \rangle$. At each round j, the egotist sends as its suggestion the time interval x_j.

The **laconic** agent will prefer not to provide too much information about itself, even at the cost of having the meeting scheduled at a less preferred time interval. The laconic strategy is to send a suggestion from the set of suggestions already made by others in previous rounds, whenever possible. At round j if S^j is the set of suggestions made until the previous rounds, and $X = \langle x_1, x_2 \ldots x_k \rangle$ is the set of ordered free intervals for the agent, and R^{j-1} is the set of suggestions the agent already offered in previous rounds, then:

- if $X \cap (S^j - R^{j-1}) \neq \emptyset$, the laconic agent will suggest the $x_i \in X \cap (S^j - R^{j-1})$ with the highest rank in X
- if $X \cap (S^j - R^{j-1}) = \emptyset$, the laconic agent suggests the $x_i \in X - R^{j-1}$ with the highest rank in X.

Thus, the laconic will avoid sending new information about itself to the negotiation, but if new information must be provided, it will provide the information that it is best for its purposes.

The **deceiving** agent behaves as the laconic agent, but when new information must be provided, it will choose a random new suggestion from its list of possible time intervals (as opposed to the laconic which will choose its best time interval as the new suggestion). Using the same notation as for the laconic,

- if $X \cap (S^j - R^{j-1}) \neq \emptyset$, the agent will suggest any randomly select $x_i \in X \cap (S^j - R^{j-1})$.
- if $X \cap (S^j - R^{j-1}) = \emptyset$, the agent suggests any randomly selected $x_i \in X - R^{j-1}$.

The deceiving agent then will provide the least information about itself allowed by the protocol. Even if a new suggestion is made by the deceiving agent, one will not be able to derive any preference information about it.

4 Simulation Results

The protocols were implemented in a simulated environment in order to measure their efficacy according to our metric. The full information and approval protocols are optimal and thus no simulations were run. The simulation results refer to the voting and suggestion protocols.

The simulations parameters were:

- the number of participants,
- the number of solutions for the scheduling problem,
- the range of preferences for each agent.

The number of participants is clearly an important parameter - the larger the number of participants, the harder it is the negotiation to schedule a meeting. Furthermore, the number of participants is the only parameter the user can see when using the system. But the number of participants conceals different factors which may be independent from each other. For example, the larger the number of participants, the smaller is the number of possible free intervals to schedule the meeting. On the other hand, more participants will likely increase the spread of preferences for each possible free time, which by itself would make the negotiation harder. Thus besides the number of participants, we decided to evaluate the effect of these two independent factor in the efficacy of the meeting.

The second simulation parameter is the number of possible solutions for the problem, that is, the size of Poss(m), independent of the number of participants. The third simulation parameter control the range of the preference values for each free time. The preferences are randomly selected from the interval from 1 to the specified range, for each free time and for each agent.

It is interesting to notice that if all participants have the same preference for each time slot then the voting protocol, as well as the egotistic strategy in the suggestion protocol are optimal. If all agents have the same preference for all possible free times, the standard deviation of the preferences is zero and the metric reduces to the average of preferences. But since the preferences are the same, the time interval with the highest preference will be the best interval to schedule the meeting. Let us call that interval x. In the voting protocol, each agent will rank x among the first group (an agent may have other intervals with the same preference). Thus x will receive the highest preference value in the host's preference approximation, and thus will be chosen by the host.

In the egotistic strategy for the suggestion protocol, each agent will suggest all of its free intervals with preference above that of x and then eventually suggest x, which will be the first suggestion that can be accepted by all other agents. The same will happen to all agents, and thus when the last one suggests x, that will be chosen as the meeting schedule.

The simulation involved 50 repetitions of creating a set of calendars with the appropriate number of solutions and number of participants. All agents would also have at least twice the number of solutions as free intervals, and each free interval was shared among at least two agents. For each participant, the preference for each time slot was randomly generated from an uniform distribution from 1 to the specific range. Simulations were caried for 5, 10, and 15 participants, for 5, 10, 20, and 30 solutions, and for 2, 5, 7, and 10 as the range of preferences.

Section 4.1 discusses the efficacy of the voting protocol. Section 4.2 discusses the efficacy of the suggestion protocol under different conditions for groups of agents that have all the same strategy. Section 4.3 discusses the suggestion protocol in the situation in which agents of different strategies are mixed in the same negotiation. This allows for the comparison of different strategies when facing each other. Section 4.4 introduces a new type of agent in the suggestion protocol, an agent that knows the other agents preferences and uses that for the "good of the group," and discusses the changes in the efficacy of the protocol when one or more such agents is added to a group. Finally section 4.5 discusses the effect of coalitions of agents that negotiate privately among themselves a good meeting time, and in the large negotiation, act as a single front.

4.1 Efficacy of the voting protocol

The voting protocol is not optimal, but it is very efficacious. The average optimization for the voting protocol was 0.98 (95% confidence interval from .0977 to 0.985). The average rank of the solution was 1.14 - only 12% of the simulations did not obtained the first best time interval as the solution achieved by the protocol. And only 3% did not achieve the second best time interval.

As for the dependence on the parameters of the simulation, figure 2 displays the changes of the mean optimization levels for the preference ranges, for the number of participants, and for the number of solutions. The vertical bars indicate the Bonferroni corrected 95% confidence interval on the mean optimization. The Bonferroni method is a multiple comparison method in which the confidence level is corrected so that statements that refer to all possible pairwise comparisons have the desired significance level. If n populations are to be compared, there are $n(n-1)/2$ pairwise comparisons. To achieve a global significance level of $1-\alpha$, pairwise comparisons are made at the significance level $1 - \frac{\alpha}{n(n-1)/2}$. For example, for the number of participant graph, each vertical bar will indicate the $1 - \frac{0.05}{3\times 2/2} = 98.3\%$ confidence interval on the average optimization. The statement that all averages are not significantly different (because their confidence intervals taken pairwise have non empty intersections) has a 95% significance level.

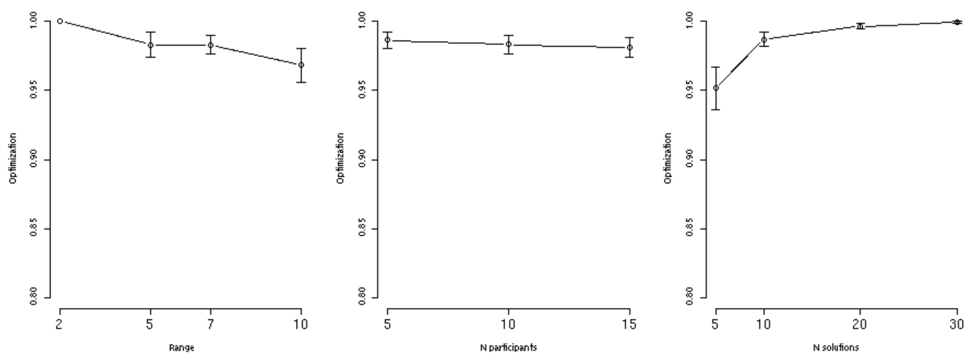

Fig. 2. Voting protocol optimization

4.2 Suggestion protocol: Homogeneous groups

We simulated groups in which all agents were egotistic, laconic, and deceiving. The group of egotistic agents achieved an average optimization of 0.797, and an average rank of 2.42. 48% of the simulations achieved the best solution, 20% the second best solution, and 12% the third best solution.

As for the dependency on the number of solutions, number of participants and range of preferences, figure 3 summarizes them. The efficacy of the egotistic strategy depends on each of the parameters. The interesting result is that if the range is too small (2), that is, the intervals are judged by the agents as either “good” or “bad” for a meeting, the quality of the schedule is poor. Which is surprising given that if the preference is the same for all agents than the egotistic strategy is optimal. The reasons for the low quality of the schedule for a range of 2 seems more related to the metric itself that to the protocol. If all intervals have only two possible preference values, there are few possible results for the group satisfaction, and the results are spread apart, and thus if the protocol does not select the best interval, the optimization level will be low. The phenomenon disappears if the range is equal or greater than 5.

As expected, the quality of the schedule decreases with the number of participants, but the difference between 10 and 15 participants is not significant. The quality increases with the number of solutions until saturation (the differences among 10, 20, and 30 solutions are not significant).

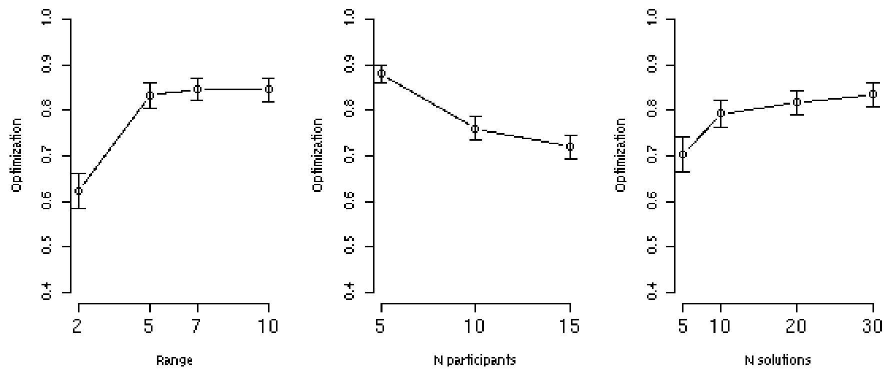

Fig. 3. Optimization level of egotistic group of agents

Groups of laconic agents performed significantly worse than the egotistic group, and groups of deceiving agents significantly worse than laconic groups. The table 1 summarizes the results for each strategy with the 95% confidence interval for both the optimization and the rank.

The figure 4 and 5 summarize the influence of the number of participants, number of solu-

Strategy	group rank				optimization		
	mean	95% confidence		median	mean	95% confidence	
		lower	upper			lower	upper
Egotistic	2.42	2.32	2.51	2	0.79	0.78	0.80
Laconic	4.64	4.45	4.83	3	0.60	0.58	0.61
Deceiving	5.54	5.32	5.76	4	0.56	0.54	0.57

Table 1
Summary of the group statistics for different strategies

tions, and range of preferences on the mean optimization level achieved for groups of laconic and deceiving agents. The results regarding the range or preferences are similar to the egotistic - a range of 2 decreases the optimization significantly, but ranges equal or above 5 have the same mean optimization. The laconic strategy is independent of the number of participants and the number of solutions, which is somewhat surprising. The reasons is that as soon as a possible schedule has been proposed, it is chosen.

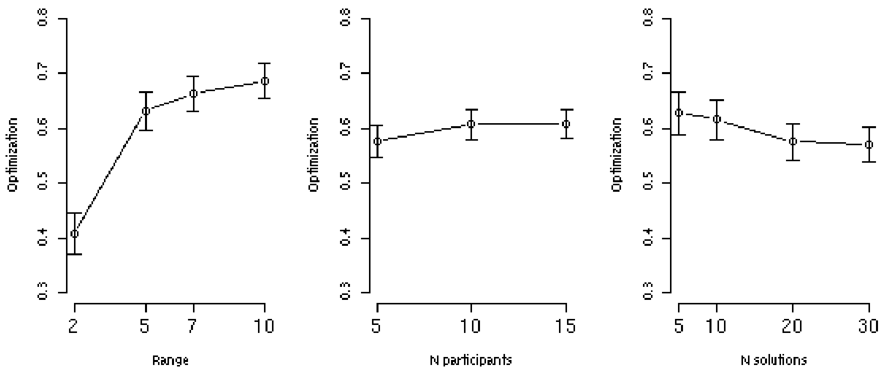

Fig. 4. Laconic agents

Regarding the user's satisfaction for homogeneous groups of egotistic, laconic, and deceiving agents, table 2 resumes those results. The confidence is also Bonferroni adjusted, so all comparisons can be made with 95% confidence. The table shows that the adjusted user

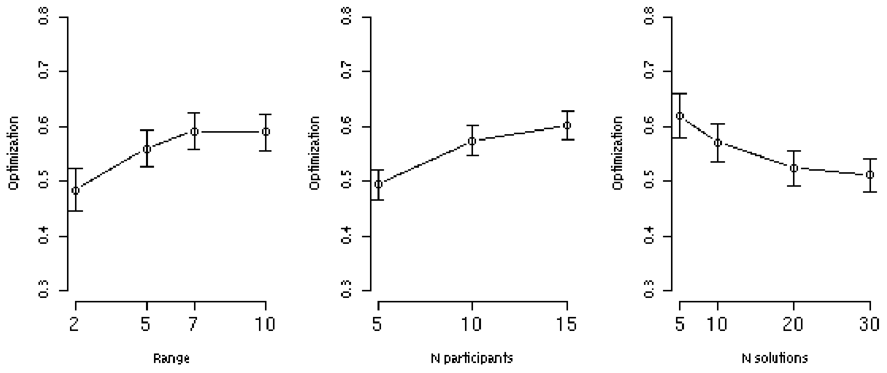

Fig. 5. Deceiving agents

satisfaction is significantly higher for egotistic agents, and that laconic agents outperform deceiving ones.

Strategy	adjusted satisfaction		
	mean	95% confidence	
		lower	upper
Egotistic	0.73	0.73	0.74
Laconic	0.70	0.69	0.70
Deceiving	0.66	0.66	0.67

Table 2
Summary of the adjusted user satisfaction for each strategy

In summary, groups of egotistic agents perform better than groups of laconic agents, from the point of view of group optimization, with an optimization of 0.78 and an average rank of 2.42, and laconic agents performed better than deceiving ones. From each agent's point of view, the egotistic strategy also results in better adjusted satisfaction (0.73) than the laconic or deceiving strategy.

We simulated a group laconic agent which contain one egotistic agent. In terms of adjusted user satisfaction, the egotistic agent has a large advantage in relation to the laconic agents. The egotistic agent achieve a satisfaction of 0.91 as opposed to a 0.69 adjusted satisfaction for the laconic agents. Figure 6 shows the satisfaction for the first 5 agents, where the fifth agent is the egotistic one.

The egotistic agent aggressively seeks its own interest while the laconic will not, and thus the resulting first schedule will likely be very good for the egotistic, and not so good for all the laconic agents. The same is true for an egotistic agent among deceiving agents.

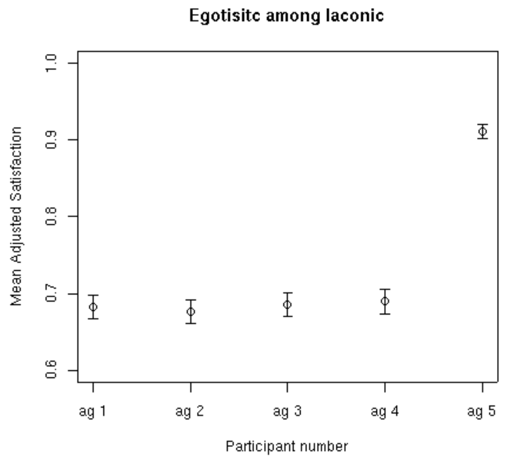

Fig. 6. An egotistic among laconic agents (5th agent is egotistic)

Thus, there seems to be very little incentive to use the laconic or deceiving strategies, besides the fact that they provide less information about its user's free time and preferences than an egotistic strategy.

A **learning** agent will, through previous negotiations with the other agents, learn the other agents's preference profiles or even strategies (for example [14]). That should be contrasted with an adaptive agent which tries to learn his own user's preferences. Of course, the frontier between the two types is somewhat fuzzy - some adaptive agents may just use the traces of all previous interaction of another agent to learn that agent's preferences, for example [15]. If the learning agent knows that a particular user prefers morning meetings, or even that he prefers meetings at 9am with preference 8, at 10am with preference 9, and so on, how can this information be used? We define an **altruistic** agent as an upper limit to a learning agent. The altruistic agent *knows* all the other agents' preferences, but not their free times, and will use this knowledge on to the benefit of the group. An altruistic agent will rank its own free time intervals according to the metric being used to evaluate the quality of the schedule. Again we run 300 simulations with 10 agents, 10 solutions, range of 10, and varying number of altruistic agents from 0 to 3.

Figure 7 shows that there is no significant differences among a group of no altruistic agent to groups up to 3 altruistic agents. Thus the altruistic agents "acting for the greater good" do not make any difference on the optimization level achieved. On the other hand there is a personal loss in behaving altruistically, the adjusted user satisfaction for an altruistic agent is 0.56 ± 0.04, which should be compared with the egotistic agent, which has a satisfaction of 0.69 ± 0.04.

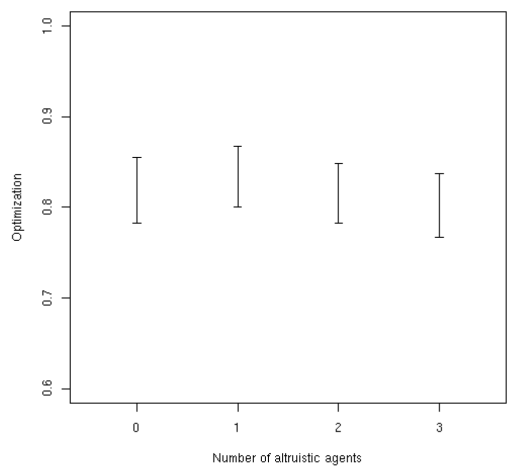

Fig. 7. Optimization levels for groups with altruistic agent

4.5 Suggestion protocol: Coalitions

In some situations, a subgroup of the participants in a meeting have no problem in sharing information among themselves, but have restrictions with sharing it with participants outside the subgroup. This subgroup may form a **coalition** that negotiates among themselves the scheduling of the meeting in a low privacy mode, and behave in a unified way in the general negotiation, in the hope that a better meeting for the subgroup would be scheduled. Coalitions seems reasonable if the negotiation is being made in the no information mode, and a subgroup trust each other well enough to negotiate in full information or preference modes, which would guarantee the subgroup's optimal scheduling.

We tested coalition formation in a group of egotistic agents working in no information mode. Among the egotistic agents, we created coalitions of 2, 3, and 4 agents which previously negotiated in full information mode, and during the main group negotiation acted in an uniform way. Each coalition agent used the coalition's free intervals and preferences as its own free intervals and preferences.

The presence of a coalition does not change significantly the optimization level achieved by the group, when compared with a set of egotistic agents with no coalition. The left plot in Figure 8 shows the optimization levels for a group with no coalition, and with a coalition of 2, 3, and 4 agents. The right plot in Figure 8 shows the results for the personal ranking of the first agent, when it belongs to no coalition, and when it belongs to a coalition of 2, 3, and 4 agents. The coalition members fare the same as the non-coalition members in terms of the personal ranking of the resulting meeting.

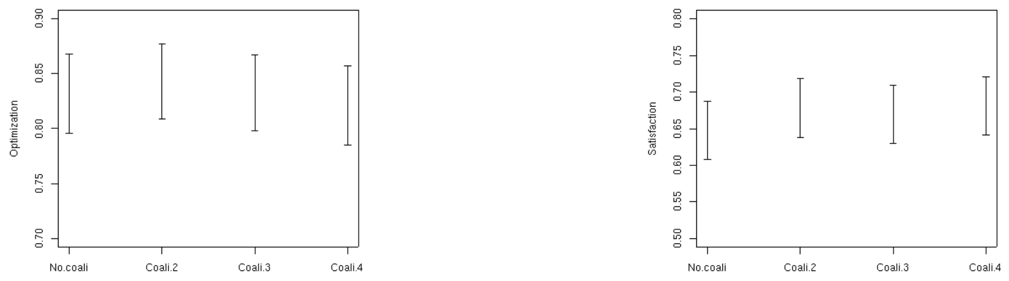

Fig. 8. Optimization and Adjusted user satisfaction for coalitions

This result shows that there is no incentive to form coalitions. The reason why coalitions are ineffective in the suggestion protocol is that the interval that is chosen is the first one that is agreed upon by all agents. There is no measure of how strong an agent agree with that time interval, nor how many agents “strongly or weakly” agree with it. Thus, the agents in a coalition, since they behave as a block, only count as a single agent in the negotiation process.

5 Related Work

The work by Garrido and Sycara [11] has strong similarities with our own. Each agent knows the preferences of its user, and negotiate with the other agents for a time interval in which the meeting can be scheduled. As a further enhancement to the concept of preferences, they propose a second preference information that allows the preferences to relax as the number of rounds in the negotiation increases.

The basic protocol is as follows:

- a round coordinator (equivalent to our host) is chosen at random from the participants.
- the coordinator proposes a time interval for the meeting.
- each agent accepts or refuses the interval.
- if all agents accept it, the meeting is scheduled in that time interval; if not all accept it, a new round coordinator is chosen and the process starts again, with the possible change in the preferences due to the relaxation process.

An important characteristic of their work is that other attributes besides the start time of the meeting are negotiated. The authors define three attributes that can be negotiated: the date of the meeting, the start time within the date, and the duration. The agents must agree on these three attributes to schedule the meeting. The metric of quality of the meeting is the average of the preferences. Another metric of interest in the paper is the number of rounds to achieve a schedule.

The authors discovered that the quality of the meeting increases if all agents have the same preference profile. We discussed in the beginning of section 4 that if all preference profiles are the same, our protocol achieves the best solution, but we have some contradictory results that if the preferences profiles are close together, the suggestion protocol results in worse quality schedule.

In a second work [17], the authors discuss the possibility of agents learning the behavior of the other agents, and they use terms like *egotistic* and *altruistic* but there is no discussion on what operational consequences such attitudes would have to the agents. No results are presented regarding the effect of learning the others' behaviors.

The work by Sen and Durfee [18,10] is mainly concerned with the problem of simultaneously scheduling multiple meetings among many agents, and the interference that one meeting causes on the scheduling of the others. The authors propose a general protocol: the host sends out a meeting time proposal, collects the answers, and if there is no convergence, the host starts a new round. To further define the protocol, the authors define three sets of parameters, or *heuristic strategies* as they called them:

- announcement strategies: whether the host suggest only one or three possible time slots for the meeting, respectively the *best* and *good* strategies.
- bidding strategies: whether each agent answers the host with a yes/no for each proposal (the *yes/no* strategy), or whether the agents must propose an alternative to the host's proposal if it does not accept it (the *alternative* strategy).

- commitment strategies: whether each agents blocks the time slots proposed (by the host and by itself) and only unblocks them if the negotiation fails (the *committed* strategy), or whether the no such commitment is made (*non-committed* strategy). This is only relevant for the case of multiple simultaneous negotiations.

This research has similarities with ours: our suggestion protocol, using their language, assumes the *best* announcement strategy, and the bidding strategy as *alternative*. But there are significant differences: there is no concept of preferences in their research (except in the sense that the algorithm tries to schedule the meeting as early a possible, and so there is a global preference for scheduling meetings sooner than later). The metrics of interest in the paper are whether the meetings could be scheduled or not, and the number of rounds until the scheduling. The authors present both analytical and simulation analysis of the various alternatives for heuristic strategies.

In [9] the concept of preferences is added to the basic model. Their preference model assumes that preferences can be constructed from the composition of orthogonal characteristics such as day of the week, time of the day, participants, topic, location of the meeting, and so on. The user attributes preferences to these values and the system computes the ranking of preferences *for a single user*. These preferences are not used to measure the resulting meeting, only to help in selecting the user's answer to the host's proposal. No simulation or analytical results are presented.

A recent work by Crawford and Veloso [20] discusses a protocol in which there are no preferences, and the agents make public their free times, but agents are engaged in may meeting scheduling concurrently, so even if an agent declared before that a time interval was free, it may no longer be free because another meeting was scheduled at that time. The metric of interest is the number of messages exchanged and the time to reach an agreement.

The work by Ephrati, Zlotkin, and Rosenschein [19] propose an external mechanism that discurages users to be the one that "causes problems" in scheduling the meeting.

6 Discussion and future work

6.1 Implementation

6.2 Limitations of this work

The protocols presented, although centered on the issues of preferences and privacy, do not protect the agent against malicious attacks by other agents. Nor does it protects the agent form unwise decisions regarding the privacy mode. As an example of the latter, an agent that must keep its user's free time private should not accept an internal host, nor a public negotiation. Each of these modes would allow at least the host to partially know the non-free time slots by verifying which time intervals were not accepted by the agent in the approval protocol. Similarly if n-1 of the agents are willing to share information among themselves, even though it is not worth for them to form a coalition, they may infer about the free times, preferences, and even strategy of the last agent.

6.3 Conclusions

This paper presented some results on a multi-agent based scheduling system. The protocols for different levels of privacy were presented. The information the user is willing to make public is the central aspect of the protocols. The protocols for the cases where the group has no problem in sharing all information (free time and preferences) or just the preference profiles, are optimal. For the cases where either preferences or both free time and preferences are not made public, the protocol is not optimal, but it is complete.

We presented efficacy results for the non optimal protocols. In particular for the suggestion protocol, we showed that the egotistic strategy will achieve better efficacy when compared with other strategies if they are uniformly adopted by the agents. In heterogeneous groups the egotistic strategy also fares better than non egotistic agents, regarding the adjusted satisfaction.

We showed that the suggestion protocol is coalition-free, that is, it is not useful for an agent to form coalitions with a subset of the participants and after negotiating with lower privacy with the coalition members, act as an united front against the non-coalition members. Furthermore we showed that there is no gain in learning the other users preference profiles and acting according to the resulting group preferences.

Finally, there is interesting research to be done regarding lying and rescheduling meetings. It is not clear if lying regarding free intervals would being any other consequence besides reducing the number of possible solutions. It is also not clear what are the consequences of lying about preferences in the full information and approval protocols.

References

[1] J. Grudin, Groupware and social dynamics: Eight challenges for developers, Communications of the ACM 37 (1) (1994) 92–105.

[2] A. Chapanis, J.F.Kelley, How professional persons keep their calendars: Implications for computerization, Journal of Occupational Psychology 55 (1982) 241–256.

[3] J. Grudin, L. Palen, Why groupware succeeds: Discretion or mandate?, in: Proceedings of the 4th European Conference on Computer-Supported Cooperative Work (ECSCW95, Kluwer Academic Publishers, 1995, pp. 263–278.

[4] R. Mitchell, T. M.and Caruana, D. Freitag, J. McDermott, D. Zabowski, Experience with a learning personal assistant, Communications of the ACM 37 (1994) 80–91.

[5] A. Cesta, D. D'Aloisi, Mixed-initiative issues in an agent-based meeting scheduler, User Modeling and User-Adapted Interaction 9 (1999) 45–78.

[6] M. T. Gervasio, M. D. Moffitt, M. E. Pollack, J. M. Taylor, T. E. Uribe, Active preference learning for personalized calendar scheduling assistance, in: IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces, ACM Press, New York, NY, USA, 2005,

pp. 90–97.

[7] P. M. Berry, M. Gervasio, T. E. Uribe, K. Myers, K. Nitz, A personalized calendar assistant, in: AAAI Spring Symposium Series, 2004.

[8] J. Oh, S. F. Smith, Learning user preferences in distributed calendar scheduling, in: The International Series of Conferences on the Practice and Theory of Automated Timetabling, 2004, pp. 35–49.

[9] S. Sen, T. Haynes, N. Arora, Satisfying user preferences while negotiating meetings, International Journal of Human-Computer Studies 47 (1997) 407–427.

[10] E. Durfee, S. Sen, A formal study of distributed meeting scheduling, Group Decision and Negotiation 7 (1998) 265–289.

[11] L. Garrido, K. Sycara, Multi-agent meeting scheduling: Preliminary experimental results, in: Second International Conference on Multi Agent Systems (ICMAS'96), 1996, pp. 95 – 102.

[12] A. B. Hassine, X. Defago, T. B. Ho., Agent-based approach to dynamic meeting scheduling problems,, in: Third International Joint Conference on Autonomous Agents and Multiagent Systems, Vol. 3, 2004, pp. 1132–1139,.

[13] P. Ferreira, J. Wainer, Scheduling meetings through multi-agent negotiation, in: Proceedings of the 15th Brazilian Symposium on AI (SBIA), Vol. 1952 of Lecture Notes in Computer Science, Springer Verlag, 200, pp. 126–135.

[14] E. Crawford, M. Veloso, Learning to select negotiation strategies in multi-agent meeting scheduling, in: Proceedings of the 12th Portuguese Conference on Artificial Intelligence, EPIA, Vol. 3808 of Lecture Notes in Computer Science, Springer Verlag, 2005, pp. 584 – 595.

[15] M. Gervasio, W. Iba, P. Langley, Learning user evaluation functions for adaptive scheduling assistance, in: Proceedings Sixteenth International Conference on Machine Learning, 1999, pp. 152–161.

[16] icalendar, http://www.ietf.org/rfc/rfc2445.txt.

[17] L. Garrido, R. Brena, K. Sycara, Cognitive modeling and group adaptation in intelligent multi-agent meeting scheduling, in: First Ibero American Workshop on Multi-Agents, 1996, pp. 55–72.

[18] E. Durfee, S. Sen, On the design of an adaptive meeting scheduler, in: Tenth IEEE Conference on Artificial Intelligence Applications, 1994, pp. 40–46.

[19] E. Ephrati, G. Zlotkin, J. Rosenschein, Meet your destiny: a non-manipulable meeting scheduler, in: Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, ACM Press, 1994, pp. 359–371.

[20] E. Crawford, M. Veloso, Opportunities for learning in multi-agent meeting scheduling, in: Proceedings of the AAAI 2004 Symposium on Artificial Multiagent Learning, 2004.