Available online at www.sciencedirect.com
ScienceDirect
Decision Support Systems 44 (2007) 285–297
Decision Support Systems
www.elsevier.com/locate/dss

Scheduling meetings through multi-agent negotiations

Jacques Wainer [a,*], Paulo Roberto Ferreira Jr. [b], Everton Rufino Constantino [a]

[a] Institute of Computing, State University of Campinas, Campinas, 13083-970, SP, Brazil
[b] Instituto de Ciências Exatas e Tecnológicas, Centro Universitário Feevale, Novo Hamburgo, 93352-000 RS, Brazil

Received 28 March 2007; accepted 28 March 2007
Available online 29 April 2007

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 when all agents are comfortable with sharing their preference profile and free times; b) the approval protocol when only the preference profile can be shared; c) the voting protocol when only free time can be shared; and d) the suggestion protocol if neither preference nor free time can be shared. We use 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. Results show that the voting protocol achieves the best solution 88% of the time. Simulation results for the suggestion protocol with different numbers of agents, different numbers of solutions, and different strategies are presented. The suggestion protocol is shown to be coalition-free.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Meeting scheduling; Multi-agent negotiation; Calendar

1. Scheduling meetings

Group calendars and meeting schedulers have been one of the first groupware applications [15]. Anyone who has ever tried to schedule a meeting among more than three people can appreciate the usefulness of some sort of automatic help. 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' calendar data 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 ([3] discusses that 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. To schedule a meeting, the user selects the participants and instructs his agent to negotiate a time interval with all the other participants' agents. The problem of scheduling a meeting becomes a negotiation among agents that defend their users' 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, an "important" person can be late for a meeting, can cancel a meeting with shorter notice, can be less flexible in scheduling a meeting, and so on. Thus, there are a lot of socially relevant issues related to

* Corresponding author.
E-mail addresses: wainer@ic.unicamp.br (J. Wainer), pauloroberto@feevale.br (P.R. Ferreira), constantino.everton@gmail.com (E.R. Constantino).

0167-9236/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2007.03.015

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 administrative meetings in the morning and research meetings late in the 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 problem instead of a satisfying problem: one has to find a time slot that is the best 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. 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, homogeneity of the user's satisfaction with the scheduled meeting, 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 maximizing the group satisfaction. Thus, each participant can have some assurance that the others have similar preference for the meeting as he does.

Multi-agent meeting scheduling is an active research area. Some of the work 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, [18,2,13,1,19] among others. This line of research, which we will call *adaptive* agents, is usually interested in the adaptability of the agent to its own user — the negotiation and privacy issues are not deeply dealt with. For example, [2] describes such an adaptive agent which uses a negotiation protocol in which all agents share their free times.
- research on the multi-agent aspects of the negotiation, [20,7,11,16], among others. This line of research is more concerned with defining a single negotiation protocol that allows for many simultaneous meeting negotiations, and studying the efficiency of the protocol, that is, the number of negotiation rounds it takes to achieve a solution. For example, [16] 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 in the previous round of negotiation.

This paper falls into the multi-agent negotiation approach but is different than most studies because:

- 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 in the efficiency (the time or number of messages needed to achieve the solution), and
- we use a non-standard metric to evaluate the quality of a schedule, which attempts to balance the average and the homogeneity of the participants' preferences for the schedule.

Simultaneous meeting negotiations are not considered in this paper because in experiments with the implemented system, negotiations never took more than a few seconds, and thus it is very unlikely that an agent will be called to negotiate a new meeting while a previous negotiation is taking place.

Regarding our lesser interest in the efficiency of the protocol, we assume that the agents are fully automated, and thus the negotiation is performed without the participants intervention. If the user has to intervene during the negotiation, for example, as it is expected in the system described in [2], then the number of messages and rounds are important issues — one would not like to bother the user with many interventions in the negotiation.

This paper is organized as follows. Section 2 describes the formal definitions and the metric used

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 the protocols that are not optimal. Section 5 discusses the related work in more details and Section 6 presents briefly details of the implemented system and future 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 planned to last, and
- w_b and w_e are the beginning and the end of the 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 at all. In the latter case we will say that the time slot is free. In this work, the calendar of an agent a_i is represented by the set Free(a_i) of free time slots. That is, if ts$\in$Free(a_i), then ts is a free time slot for agent a_i.

Time interval is defined as a sequence of one or more adjacent time slots. A time interval t is free, that is, $t\in$Free(a_i) if, for all time slots ts in t, ts$\in$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, both free time slots and free time intervals are referred as free time.

Each agent a_i has a preference function or utility 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 Poss($m$)$=\{t_1, t_2... t_k\}$ as the set of all possible time intervals to schedule the meeting. That is, for all $t_j\in$Poss(m), $t_j\in$Free(a_i) for all agents $a_i\in A$, the duration of each t_j is equal to l, and each t_j is between w_b and w_e.

2.1. Metric of evaluation

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

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

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:

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

The definition above is the mean of the users' preferences $\left(\frac{1}{N}\Sigma_i\alpha_i(t)\right)$ minus the standard deviation of the preferences $\left(\sqrt{\frac{\Sigma_i(...)^2}{N-1}}\right)$. This metric is our solution to the issue of the homogeneity of the user's satisfaction with the scheduled meeting, as discussed in the Introduction. The metric tries to maximize the average of the users' preferences but penalizes a large dispersion of the preferences (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:

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

Given the sequence $\langle t_1, t_2,...t_i...t_n\rangle$ of time intervals $t_i\in$Poss(m), ordered by decreasing value of the group utility, the group rank of the time interval t, denoted as $R(t)$ is defined as the rank of each time interval: $R(t_1)=1$, $R(t_2)=2,..., R(t_i)=i$.

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. The adjusted user satisfaction measures how close the scheduled meeting is to the best possible meeting time

for the particular user. An adjusted user satisfaction of 1 indicates that for that user, the protocol was maximally effective.

3. The negotiation protocols

3.1. Levels of privacy

As discussed in the Introduction, privacy is a important issue regarding calendars and meeting scheduling. It is clear that in some situations an agent will not want to share its user's free time with the other agents, and in some other situations, that user may have no problem in sharing that information, specially if that would increase the chances of scheduling a meeting in a time interval that is good for him and for the group.

Four levels of privacy or modes of interaction are defined depending on whether the users are comfortable in sharing their preference profile and free time with the host of the negotiation:

(1) full information: the user has no problem in making both his free time and preference information available to the host,
(2) preference: the user's preference profile can be shared with the host, but not the free times,
(3) free time: the user's free time information can be shared with the host, but not the preference, and
(4) no information: the user does not wish to make either the preference or the free time available.

We consider that the no information mode has the highest privacy level, followed by both the free time and the preference modes, which are incomparable between each other, and finally the full information is the lowest privacy level.

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 $\bar{a}$ the lowest level of privacy it will accept for the negotiations to schedule this meeting. The host then computes the highest privacy level that is greater or equal to all agents' requirements, and define that as the privacy level for the negotiation.

3.2. Efficacy of the negotiation protocols

This paper is 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 decide that a meeting is not possible when it really is. Of course, an optimal protocol is also complete.

Although efficiency is not the main concern of this paper, we will present results regarding the upper bound to the number of rounds and to the number of messages needed to achieve a solution. We define a *round* as a message 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. A *half-round* is used to refer to a one-way flow of messages. The protocols below will exchange free time intervals, agreement or disagreement with a proposed schedule, preferences, and so on, which will be referred as *pieces of information*, and each message will carry one of such piece of information. Finally, N denotes the number of participants and M denotes the number of time intervals in the scheduling window.

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 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 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, which 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. Approval protocol

The approval protocol is used in the preference mode. Each agent 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 submits 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: suppose that the host creates the ordered list of time intervals $\langle t_1, t_2, \ldots t_x \ldots \rangle$ by decreasing order of group utility, and that t_x is the best schedule for the meeting. That means that for all $i<x$ 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 schedule 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 below), 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.

All meeting scheduling protocols and implemented systems are susceptible to lying (see [14] for a discussion on lying in the first implemented meeting scheduling system). Our protocols do not address this issue, and as far as we are aware, the only attempt to design a negotiation protocol that discourages lying is presented in [8] which requires external mechanisms.

3.5. 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.

3.6. Suggestion protocol

The suggestion protocol is used in the no-information mode — 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 collects the suggestions, remove the information of which agent made which suggestion, and sends 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 to return just the new suggestions proposed in the previous round. Similarly, in each round, the agent only needs to send one 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^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_{ij}
    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 $\cap_i R_i^j = \emptyset$ 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

3.6.1. 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 one 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.

3.6.2. 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 time, at the expense of providing some information about itself to the host. The egotistic strategy is to rank its set of free intervals in decreasing order of preference $\langle x_1, x_2..x_k \rangle$. 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 $\langle X=x_1, x_2..x_k \rangle$ 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 $X \cap (S_j - R^{j-1}) = \emptyset$, 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 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 $X \cap (S^j - R^{j-1}) = \emptyset$, the agent will suggest any randomly selected $x_i \in X \cap (S^j - R^{j-1})$.
- 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 from 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 was run. The simulation results refer to the voting (Section 4.1) and suggestion protocols (Sections 4.2–4.5).

The simulations parameters are:

- 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 because the larger the number of participants, the harder is the negotiation to schedule a meeting. Furthermore, the number of participants is the only parameter the user can see when using the system. However, 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, the effect of these two independent factors in the efficacy of the meeting is also evaluated. The third simulation parameter controls 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.

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 a uniform distribution from 1 to the specific range. Simulations were carried out 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.

4.1. Efficacy of the voting protocol

The voting protocol is not optimal, but it is very effective. The average optimization for the voting protocol is 0.98 (95% confidence interval from 0.977 to 0.985). The average rank of the solution is 1.14. Only 12% of the simulations did not schedule the meeting on the best time interval and only 3% did not schedule it on the second best time interval.

As for the dependence on the parameters of the simulation, Fig. 1 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.

4.2. Suggestion protocol: homogeneous groups

This section presents the results for groups where all agents were either egotistic, laconic, or deceiving. The group of egotistic agents achieves 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.

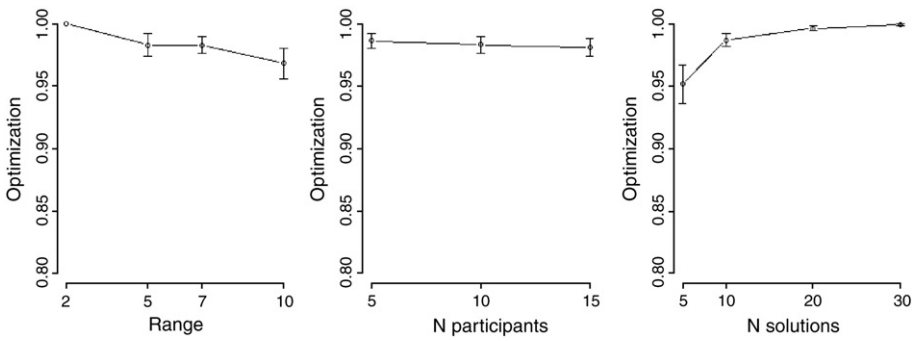

Fig. 1. Voting protocol optimization.

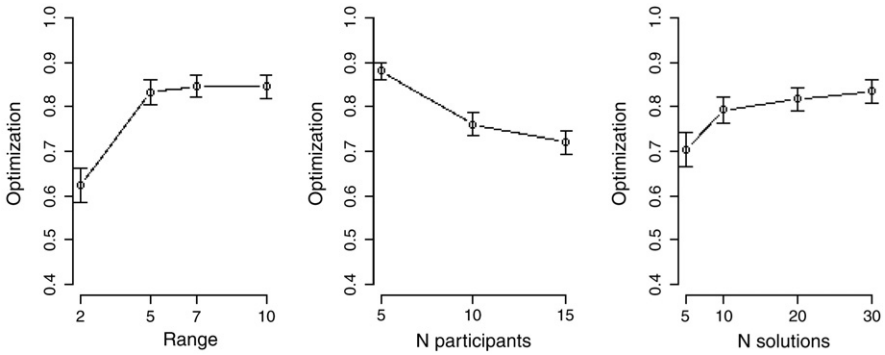

Fig. 2. Optimization level of egotistic group of agents.

Fig. 2 summarizes the effect of the number of solutions, number of participants, and range of preferences on the optimization level. 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. This 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 to be 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. If there are only few solutions (5), the quality of the solution is lower, but stabilizes for 10 or more solutions. A group of egotistic agents performs significantly better than a group of laconic agents, which in turn performs better than the deceiving agents. Table 1 summarizes the results for each strategy with the 95% confidence interval for both the optimization and the rank.

Figs. 3 and 4 summarize the influence of the number of participants, number of solutions, and range of preferences on the mean optimization level achieved for groups of laconic and deceiving agents. The results regarding the range and preferences are similar to the egotistic strategy — a range of 2 decreases the optimization significantly, but ranges of preferences equal or above 5 have the same mean optimization. The laconic strategy is independent of the number of participants and the number of solutions. The reason is that as soon as a possible schedule is proposed, it is chosen.

Table 2 summarizes the results regarding the user satisfaction for groups of egotistic, laconic, and deceiving agents. The confidence is also Bonferroni adjusted, so all comparisons can be made with 95% confidence. The table shows that the adjusted user satisfaction is significantly higher for egotistic agents, and that laconic outperform deceiving agents.

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 perform better than deceiving ones. From the agent's point of view, the egotistic strategy also results in better adjusted satisfaction (0.73) than the laconic or deceiving strategy.

4.3. Suggestion protocol: heterogeneous groups

If one considers groups with agents with different strategies, the egotistic strategy is still advantageous. For example, a single egotistic agent in a group of laconic agents achieves a mean satisfaction of 0.92 against a mean satisfaction of 0.69 for the laconic agents (the 95% confidence interval for the difference is 0.23 ± 0.01). Thus, it is better for a laconic agent to switch to the egotistic strategy.

Table 1
Summary of the group statistics for different strategies

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

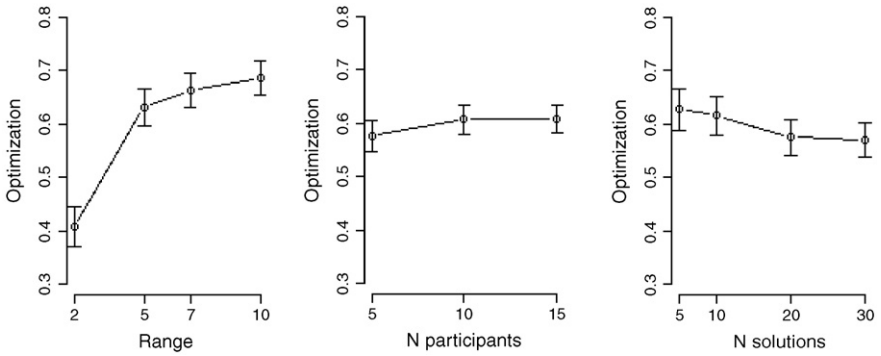

Fig. 3. Laconic agents.

On the other hand, it is not advantageous for one egotistic agent in a homogeneous group of egotistic agents to change its strategy to laconic. A single laconic agent in a group of egotistic has a mean satisfaction of 0.72 against the mean satisfaction of 0.74 for the egotistic agents. The 95% confidence interval for the difference is −0.034 to −0.006; thus the difference is small but significant.

These simulations together with the simulations for homogeneous groups show that there is 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. Thus, from now on laconic and deceiving agents will not be considered.

4.4. Suggestion protocol: Learning during the negotiations

A learning agent will, through previous negotiations with the other agents, learn the other agents' preference profiles or even strategies (for example [5]). That should be contrasted with an adaptive agent which tries to learn his own user's preferences. The frontier between the two types is somewhat fuzzy—some adaptive agents may just use the logs of all previous interaction of another agent to learn that agent's preferences, as in [12]. If the learning agent knows that a particular user prefers morning meetings, or even that he prefers meetings at 9am with preference 8, at 10 am 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 for the benefit of the group. An altruistic agent will rank its own free time intervals according to the groups utility. We performed 300 simulations with 10 agents, 10 solutions, range of 10, and varying number of altruistic agents from 0 to 3, where the remaining agents were egotistic.

Fig. 5 shows that there are no significant differences among a group of zero altruistic agent to groups up to 3 altruistic agents. Thus the altruistic agents "acting for the greater good" does 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

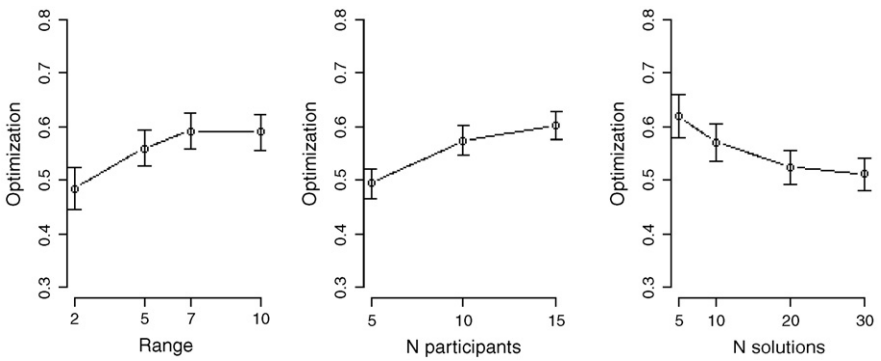

Fig. 4. Deceiving agents.

Table 2
Summary of the adjusted user satisfaction for each strategy

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

should be compared with the egotistic agent, which has a satisfaction of 0.69 ± 0.04.

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, coalitions of 2, 3, and 4 agents were created, 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 Fig. 6 shows the optimization levels for a group with no coalition, and with coalitions of 2, 3, and 4 agents. The right plot in Fig. 6 shows the results for the personal ranking of the first agent, when it does not belong to any coalition, and when it belongs to the coalitions 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.

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 as the meeting time 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 the coalition, since they behave as a block, only count as a single agent in the negotiation process.

5. Related work

As discussed in the Introduction, this work follows in the multi-agent negotiation line of research, and has some intersection with other papers in this same line.

The work by Garrido and Sycara [11,10] present a negotiation protocol that combines the suggestion and approval protocols presented in this paper. Their protocol is based on randomly selecting a host among the agents, which then proposes a time interval for the meeting. Each of the other agents accept it or not. If not all agents accept the interval, a new host is selected and a new round starts. 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 solution.

The work by Sen and Durfee [6,7] 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:

- announcement strategies: whether the host suggest only one or three possible time slots for the meeting, respectively the *best* and *good* strategies.

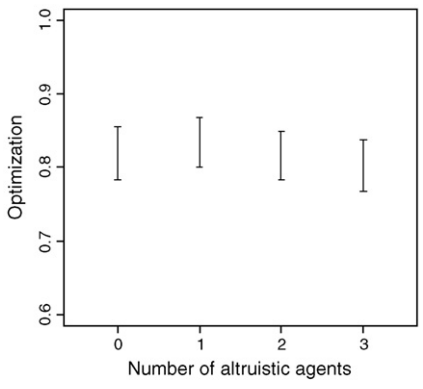

Fig. 5. Optimization levels for groups with altruistic agent.

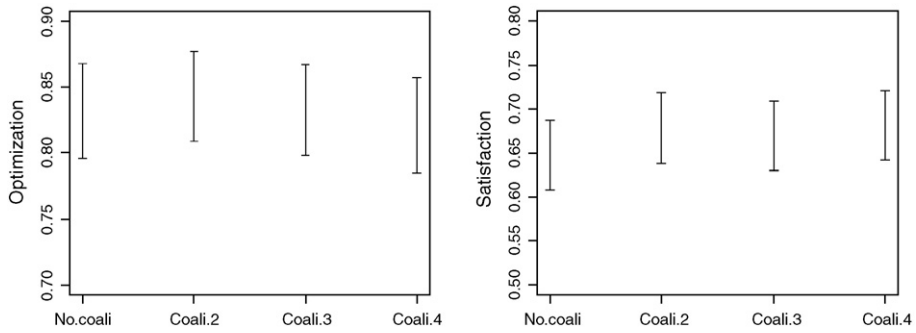

Fig. 6. Optimization and Adjusted user satisfaction for coalitions.

- 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 no such commitment is made (*non-committed* strategy). This is only relevant for the case of multiple simultaneous negotiations.

The work in [6,7] has similarities with ours: the suggestion protocol proposed in this paper, using [6,7]'s language, assumes the *best* announcement strategy, and *alternative* as the bidding strategy. But there are significant differences: there is no concept of preferences in their research and the metrics of interest are whether the meetings could be scheduled or not, and the number of rounds to achieve a solution. The authors present both analytical and simulation analysis of the various alternatives for heuristic strategies.

In [20] 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 [4] discusses a protocol in which there are no preferences, and the agents make public their free times, but agents are engaged in many 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 interval. The metric of interest is the number of messages exchanged and the time to reach an agreement.

This paper is an extension of previous work [9] by two of the authors. The protocols were first defined in that work, but the metric of evaluation used is different. Some of the strategic variations for the suggestion protocol are also not present in [9].

6. Discussion and future work

6.1. Implementation

A prototype of the system proposed here was developed in C++ and TK/TCL for Linux workstations. The main program is the personal agent, or PA, which implements both the host and the agent described in the paper. A version of the program with just the host component can also be generated.

The PA also deals with issues not discussed in this paper. For example, besides the privacy levels discussed in the paper, the PA may request that the host for the negotiation is not one of the agents involved, and a protocol to propose hosts was developed.

The PA reads in the user calendar data, stored in the iCalendar format [17], and a preference data file which associates to each meeting type and to each time slot a preference. The PA also reads in a rule file which contain privacy rules and other negotiation parameters, stated as IF/THEN rules. In this file the user specifies the email addresses of the people with whom he is willing to negotiate in full information mode, the ones with whom he is willing to negotiate in free time mode, and so on. The file defines a default privacy mode which will be used for the participants for which there is no specific privacy rule.

The implemented system also deals with cancellation and rescheduling of meetings. Each participant can cancel or withdraw his/her participation in an already booked meeting, and the owner of the meeting, that is

the participant that called the meeting, has the power to cancel the meeting altogether.

These actions open up the possibility for rescheduling meetings: the cancellation of a meeting allows each of the would be participants to reschedule some other meetings because now there is more (or at least a different set of) free time slots available. In the case of withdraw, the remaining participants of the meeting can try to reschedule it, in the hope that without the constraints brought in by the participant that canceled his participation, the new scheduled time may be better for the remaining participants.

A protocol to renegotiate a meeting was developed, but no simulation on it was run. The effect of rescheduling a meeting depends on the meetings that were scheduled after it, so that, at rescheduling, the set of free time intervals is different. It is yet unclear how to set up experiments that would allow to extract any useful statistical information on the advantages or not of renegotiating a meeting.

6.2. 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 in a lower privacy mode 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 bring 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] Pauline Berry, Melinda Gervasio, Tomas Uribe, Karen Myers, Ken Nitz, A personalized calendar assistant, AAAI Spring Symposium Series, March 2004.
[2] Amedeo Cesta, Daniela D'Aloisi, Mixed-initiative issues in an agent-based meeting scheduler, User Modeling and User-Adapted Interaction 9 (1999) 45–78.
[3] Alphonse Chapanis, J.F. Kelley, How professional persons keep their calendars: implications for computerization, Journal of Occupational Psychology 55 (1982) 241–256.
[4] Elisabeth Crawford, Manuela Veloso, Opportunities for learning in multi-agent meeting scheduling, Proceedings of the AAAI 2004 Symposium on Artificial Multiagent Learning, 2004.
[5] Elisabeth Crawford, Manuela Veloso, Learning to select negotiation strategies in multi-agent meeting scheduling, Proceedings of the 12th Portuguese Conference on Artificial Intelligence, EPIA, Lecture Notes in Computer Science, vol. 3808, Springer Verlag, 2005, pp. 584–595.
[6] Edmund Durfee, Sandip Sen, On the design of an adaptive meeting scheduler, Tenth IEEE Conference on Artificial Intelligence Applications, 1994, pp. 40–46.
[7] Edmund Durfee, Sandip Sen, A formal study of distributed meeting scheduling, Group Decision and Negotiation 7 (1998) 265–289.
[8] Eithan Ephrati, Gilad Zlotkin, Jeffrey Rosenschein, Meet your destiny: a non-manipulable meeting scheduler, Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, ACM Press, 1994, pp. 359–371.
[9] Paulo Ferreira, Jacques Wainer, Scheduling meetings through multi-agent negotiation, Proceedings of the 15th Brazilian Symposium on AI (SBIA), volume 1952 of Lecture Notes in Computer Science, Springer Verlag, 2000, pp. 126–135.
[10] Leonardo Garrido, Ramon Brena, Katia Sycara, Cognitive modeling and group adaptation in intelligent multi-agent meeting scheduling, First Ibero American Workshop on Multi-Agents, 1996, pp. 55–72.
[11] Leonardo Garrido, Katia Sycara, Multi-agent meeting scheduling: preliminary experimental results, Second International Conference on Multi Agent Systems (ICMAS'96), 1996, pp. 95–102.
[12] Melinda Gervasio, Wayne Iba, Pat Langley, Learning user evaluation functions for adaptive scheduling assistance, Proceedings Sixteenth International Conference on Machine Learning, 1999, pp. 152–161.
[13] Melinda Gervasio, Michael Moffitt, Martha Pollack, Joseph Taylor, Tomas Uribe, Active preference learning for personalized calendar scheduling assistance, IUI '05: Proceedings of the 10th International Conference on Intelligent User Interfaces, ACM Press, New York, 2005, pp. 90–97.
[14] Jonathan Grudin, Why CSCW applications fail: problems in the design and evaluation of organizational interfaces, Proceedings of the 1988 ACM Conference on Computer-supported Cooperative Work, ACM Press, 1988, pp. 85–93.
[15] Jonathan Grudin, Groupware and social dynamics: eight challenges for developers, Communications of the ACM 37 (1) (1994) 92–105.
[16] Ahlem Hassine, Xavier Defago, Tu Ho, Agent-based approach to dynamic meeting scheduling problems, Third International Joint

Conference on Autonomous Agents and Multiagent Systems, vol. 3, 2004, pp. 1132–1139.
[17] icalendar. http://www.ietf.org/rfc/rfc2445.txt.
[18] Tom Mitchell, Rich Caruana, Dayne Freitag, John McDermott, David Zabowski, Experience with a learning personal assistant, Communications of the ACM 37 (1994) 80–91.
[19] Jean Oh, Stephen Smith, Learning user preferences in distributed calendar scheduling, The International Series of Conferences on the Practice and Theory of Automated Timetabling, 2004, pp. 35–49.
[20] Sandip Sen, Thomas Haynes, Neeraj Arora, Satisfying user preferences while negotiating meetings, International Journal of Human–Computer Studies 47 (1997) 407–427.

Jacques Wainer is an associate professor at the Institute of Computing in the State University of Campinas (UNICAMP), Brazil. His academic interests and publications are in the areas of collaborative computing, artificial intelligence, and medical informatics. He is also a visiting professor at the Department of Medical Informatics at the Federal University of Sao Paulo (UNIFESP). Dr. Wainer has consulted for many companies in the area of workflow systems and artificial intelligence.

Paulo R. Ferreira Jr. is a PhD Student at Institute of Informatics in the Federal University of Rio Grande do Sul (UFRGS), Brazil. His academic interests and publications are in the areas of artificial intelligence and collaborative computing. He is also an associate professor at Feevale University Center (Feevale).

Everton Rufino Constantino is a computer science master student at the Institute of Computing in the State University of Campinas (UNICAMP), Brazil. He is currently researching time series forecast using ensemble methods.