# On the design of an adaptive meeting scheduler *

Sandip Sen
Dept of Mathematical & Computer Sciences
University of Tulsa
600 South College Avenue
Tulsa, OK 74104
sandip@kolkata.mcs.utulsa.edu

Edmund H. Durfee
Department of EECS,
University of Michigan
1101 Beal Avenue
Ann Arbor, MI 48109
durfee@engin.umich.edu

**Abstract**

*We present design considerations for an automated meeting scheduling agent that processes meeting requests on behalf of its associated user. In our formulation of the meeting scheduling problem, distributed meeting scheduling agents, one per user, intelligently exchange information with each other to schedule meetings without compromising user-specified constraints. In this paper, we first enumerate various strategies we have investigated to focus distributed negotiation between scheduling agents. Next, we demonstrate the necessity for such a scheduler to be adaptive in its choice of options for the various strategy dimensions, so that it can perform effectively over time. In order to build an adaptive scheduler that can effectively choose from available strategy options, we develop quantitative performance estimates of these options using detailed probabilistic analysis. Results from these analyses are used to provide guidelines to choose the most appropriate strategy combination given current environmental conditions and local problem-solving states.*

## 1 Introduction

Our research involves developing automated software agents to handle some of the information processing needs of humans in organizations. In particular, we have been working on the design of an automated meeting scheduling agent that can schedule meetings on behalf of associated users [4, 6]. The agents preserve privacy requirements of users, and reveal only restricted amount of information about calendar states, as and when required, to schedule meetings with other users. In this paper, we outline the design of an intelligent meeting scheduling agent that can adapt its choice of heuristic strategies for negotiation to suit the current problem-solving environment. Our design is based on the precise analytical evaluation of alternative strategy choices. We present the probabilistic analysis used by agents to choose between strategy options, and illustrate the benefit of adaptive scheduling by using example problem scenarios. Our work promises to provide efficient automated meeting scheduling capabilities that are not available with the current top-of-the-line calendar management software [8].

## 2 Distributed Meeting Scheduling

The agents in a Distributed Meeting Scheduling (DMS) system exchange relevant information to build local schedules that fit into a globally consistent schedule. To facilitate information exchange, the agents need a common communication protocol for negotiating over meeting times. For our agents, we have chosen to adapt the multistage negotiation protocol [2], which is a generalization of the contract net protocol [7]. In our protocol, each meeting has a particular agent who is responsible for it, called the host. The host contacts other attendees of the meeting (who are called invitees) to announce the meeting, and collects bids (availability information). This process could be repeated several times before a mutually acceptable time interval is found, or it is recognized that no such time interval exists. Other meetings could be undergoing scheduling concurrently; in general, an agent can simultaneously be involved in scheduling any number of meetings, acting as a host for some and an invitee for others.

How well this protocol performs in efficiently converging on good schedules is strongly impacted by heuristic strategies about what information to exchange and how to model tentatively scheduled meetings. In order to decide what to propose for a meeting, the agents have to search their calendars in a systematic manner using some appropriate search bias. Strategies for communication must balance demands for privacy (which lead to exchanging less information) with demands for quickly converging on meeting times (which can be sped up by exchanging more information). Strategies for modeling tentatively scheduled meetings can range from blocking off tentative time(s) for a meeting unless and until the arrangements fall through, to ignoring tentative commitments about a meeting when scheduling other meetings. We have embarked on research to develop, analyze, and verify a formal model of DMS to formulate rigorous, quantitative predictions of the performance of the following

*This research has been sponsored, in part, by the National Science Foundation under Coordination Theory and Collaboration Technology grant IRI-9015423, and by a grant from Bellcore.

types of heuristic strategies:

**Search Biases** determine the order in which the calendar space is searched to find acceptable time intervals for a meeting. We have considered **linear early** (**LE**) where agents try to schedule a meeting as early as possible, and **hierarchical** (**H**), where agents build a temporal abstraction hierarchy over the calendar space. At each node in the hierarchy, agents keep a record of the number of intervals of different lengths free below that node in the hierarchy. The calendar space lends itself to a very natural hierarchy of hours, days, weeks, etc., and the agents participating in a meeting can first identify a good week to meet in, then identify a good day within that week, and finally an actual interval within that day. Given a meeting of some particular length to schedule, the host obtains information about the invitees regarding how many intervals of that length are open at each node (e.g., at each week) at the highest level of the hierarchy. The host uses this information to calculate the probability of scheduling the meeting under each of the nodes, ranks the nodes, elaborates the best one, and proceeds to repeat the process for the next level of the hierarchy under the elaborated node.[1] Backtracking occurs if a particular portion of the ground level being elaborated contains no solution to the scheduling problem.

**Announcement Strategies** determine how a meeting is announced, and usually involve proposing some number of possible times. We specifically consider the options called **best** (where only the best meeting time from the host's perspective, ranked by some heuristic like being the earliest, is communicated) and **good** (where several times preferred by the host, 3 by default, are communicated).

**Bidding Strategies** determine what information an invitee sends back based on an announcement. We consider the options called **yes_no** (where an invitee says yes or no to each proposal sent by the host) and **alternatives** (where an invitee proposes other time(s) when it can meet).

**Commitment Strategies** are **committed** (when a time is proposed by a host or invitee agent, that agent tentatively blocks it off on its calendar so no other meetings can be scheduled there) and **non-committed** (times are not blocked off until full agreement on a meeting time is reached).

## 3 Adaptive scheduling

Our analysis of various strategy options [5] show that no one strategy combination dominates another over all circumstances. Changing environmental factors like system load, organization size, etc., can produce a change in the strategy combination that will produce the best results on any given performance metric. If there is a way to predict the best strategy combination for a given performance metric and given environmental and system conditions, we would like our automated scheduler to take advantage of that. Such a scheduler would be adaptive to changes in the system and the environment, providing us with the most desirable performance as measured by certain performance metrics.

In the following, we will first illustrate the benefits of adaptive scheduling by describing how an adaptive search bias technique can improve the performance of a scheduler over a static choice of the search bias. We then discuss the choice of different announcement, bidding, and commitment strategies. We note that the different strategy dimensions are not mutually independent in that certain options along one dimension combine more meaningfully with certain options from another dimension. This leads us to the development of a step by step procedure of choosing different strategy options given a snapshot of the meeting scheduling scenario which can be used by an adaptive scheduling agent every time it attempts to schedule a meeting.

## 4 Adaptive search bias

We believe that it is to our advantage to build schedulers that choose between alternative search biases based on the current context, rather than using a default option under all circumstances.

For example, if the scheduling agents were all using the **H** search bias, their schedules would be evenly dense, and all parts of the calendars would be equally likely to schedule a new meeting. At this point, one can save on the number of iterations by switching to a **LE** search bias. The latter would not incur the iterations required by the hierarchical search bias to traverse down the inner nodes of temporal abstraction of the calendar space, and would focus search on a part of the space that is just as good as any other portion.

If the **LE** search bias is used for a while to schedule meetings, there will be considerable density variations along the length of the calendar; when a meeting is being scheduled for a large number of attendees, such variations can combine to give very different success probabilities at different parts of the calendar. Additionally, if the host is more free than invitees, and the latter are using a **yes_no** bidding strategy, a savings can be obtained by using the hierarchical search bias (see Section 4.3 for examples). Hence, after a while there will be sufficient mismatch between the schedules of agents to warrant the switch back to the hierarchical search bias. The selling point of adaptive search bias is that the scheduling agent can choose the most appropriate bias for any meeting based on the current states of the attendee calendars.

To help decide between alternative search bias techniques, the following probabilistic analysis will assume the availability of the density profile characteristics (or DPCs, which depict the variation of meeting densities over the length of agent calendars) for the desired meeting length for the attendees of the meeting.[1]

### 4.1 Scheduling probability with linear early search bias

In this section, we calculate the expected number of iterations taken by agents to schedule a meeting using the linear early search bias. We assume that the total number of free intervals of any given length is

[1] While some of this information could be obtained indirectly, we assume in this paper that agents will directly communicate about the densities of various intervals, with a resulting decrease in privacy.

known for each of the days of the calendar of the agents attending a meeting. Let $possible(l, \mathcal{L}) = \mathcal{L} - l + 1$, be the maximum possible number of intervals of length $l$ open on a calendar day of $\mathcal{L}$ hours (this happens when no meeting has been scheduled on the day). Let $available(i, j, l, \mathcal{L})$ be the number of intervals of length $l$ currently free in the $j$th day of agent $i$'s calendar, where each day is of length $\mathcal{L}$.

Let us calculate the probability of scheduling the meeting on the first day (or any arbitrary day). To simplify the expressions below, we introduce the following notations: $K = possible(l, \mathcal{L})$, $A$ agents numbered $1, \ldots, A$ are the invitees to the meeting, $n_x = available(x, 1, l, \mathcal{L})$ are the number of intervals of desired length open in the $x$th attendee's calendar on the chosen day. The host of the meeting will propose free intervals from its calendar one at a time (assuming the **best** announcement strategy) until an interval is found which is free in every invitee's calendar. We will calculate $P_i$, the probability that exactly $i$ iterations are required to schedule the meeting. This means that the $i$th interval proposed was free in every invitee's calendar and, for all of the previous $i-1$ proposals, the proposed interval was not available in the calendar of at least for one invitee. Let $Pr(i)$ denote the probability that the $i$th interval is free on the calendar of every invitee, and $Pr(\overline{i}) = 1 - Pr(i)$ denote the probability that the $i$th interval is occupied in the calendar of at least one invitee. Now,

$$\begin{aligned} P_i &= Pr(i, \overline{i-1}, \ldots, \overline{1}) \\ &= Pr(i) \times \\ &\quad \{1 - Pr(i-1|i)\} \times \\ &\quad \{1 - Pr(i-2|i, \overline{i-1})\} \times \\ &\quad \vdots \\ &\quad \{1 - Pr(1|i, \overline{i-1}, \ldots, \overline{2})\}. \end{aligned}$$

There are $i$ multiplicative factors in the expression for $P_i$. The $j$th factor in the expression, where $j > 1$, is of the following form (see [3] for the proof)[2]:

$$1 - Pr(i-j+1|i, \overline{i-1}, \ldots, \overline{i-j+2}) = \frac{\sum_{y=0}^{j-1}(-1)^y \binom{j-1}{y} \dfrac{\prod_{x=1}^{A}\binom{K-y-1}{n_x-y-1}}{\prod_{x=1}^{A}\binom{K}{n_x}}}{\sum_{y=0}^{j-2}(-1)^y \binom{j-2}{y} \dfrac{\prod_{x=1}^{A}\binom{K-y-1}{n_x-y-1}}{\prod_{x=1}^{A}\binom{K}{n_x}}}.$$

The denominator of the $j$th term is the same as the numerator of the $(j-1)$th term $\forall j \in \{2, \ldots, i\}$. Hence, $P_i$ is equal to the numerator of the last of the $i$ multiplicative factors in its expression. The probability that the $i$th proposal is the first to find the same interval open in the calendar of all $A$ invitees is thus:

$$P_i = \sum_{j=0}^{i-1}(-1)^j \binom{i-1}{j} \frac{\prod_{x=1}^{A}\binom{K-j-1}{n_x-j-1}}{\prod_{x=1}^{A}\binom{K}{n_x}}. \quad (1)$$

Using this equation, we can find out the probability of the host succeeding in scheduling the meeting on any of the iterations proposing meeting on the first day. Suppose the host agent is numbered 0 and it has $n_{0,1}$ intervals open on the first day of its calendar. So, the probability of the host scheduling the meeting on the first day is $P(D_1) = \sum_{i=0}^{n_{0,1}} P_i$, where $P_i$ is as given in equation 1. Let the failure of scheduling the meeting on the first day be $Q(D_1) = 1 - P(D_1)$. The probability that the meeting is scheduled by the $i$th proposal on day 2 (requiring a total of $n_{0,1} + i$ to be proposed) is $Q(D_1) * P_i$. In general, the probability of scheduling the meeting by the $i$th proposal on the $j$th day (requiring a total of $i + \sum_{k=1}^{j-1} n_{0,k}$ intervals to be proposed) is given by $P_i * \prod_{k=1}^{j-1} Q(D_k)$. This set of probabilities constitutes a probability mass function for the random variable corresponding to the number of intervals to be proposed to schedule the required meeting given the current states of the calendars of all attendees. This is identical to the probability mass function of the random variable corresponding to the number of iterations taken by the **best** announcement strategy to schedule the meeting (since this strategy proposes one interval per iteration). If the host uses the **good** announcement strategy, proposing $N$ intervals per iteration, the probability of scheduling the meeting in $i$ iterations is given by $Pr(i, N) = \sum_{j=N(i-1)+1}^{N*i} Pr(j, 1)$, where $Pr(j, 1)$ is the probability that the **best** announcement strategy required $j$ iterations to schedule the meeting.

## 4.2 Scheduling probability with hierarchical search bias

In this section, we calculate the expected number of iterations taken by agents to schedule a meeting using the hierarchical search bias. Let us assume $P$ agents (numbered $1, \ldots, P$) are involved in scheduling a meeting of length $l$, and they have constructed identical temporal abstraction hierarchies over the base calendar space (linear ordering of calendar hours). For any internal node in the hierarchy, we will calculate the probability that one or more common intervals are free in the base space of the calendars under that node, for every attendee of the meeting.

Let $x$ be the node in question. Since, the hierarchies formed by the agents are identical, every agent has $a(l, x)$ intervals of length $l$ below this node of its abstraction hierarchy (calculation of $a(l, x)$ is simple; for now we assume that these invariant numbers are available from a lookup table). We assume that the host has also collected the number of intervals of length $l$ currently free under node $x$ for each of the agents. For agent $i$, let this number be $f_i(l, x)$.

[2] In the following and other calculations in this paper, we assume $\binom{a}{b} = 0$, if $b < 0$ or if $a < b$.

Let $A_i, \forall i \in \{1, \dots, a(l,x)\}$ be the event that the $i$th (chronologically) of the $a(l,x)$ intervals is free in every attendees calendar. The probability that at least one of the events $A_1, \dots, A_{a(l,x)}$ occurs can be calculated as follows. Algebraic counting techniques based on an inclusion-exclusion argument [1] can be used to show that, for $n$ such events we have:

$$\begin{aligned} & Pr(A_1 \vee A_2 \vee \dots \vee A_n) \\ & = Pr(A_1) + \dots + Pr(A_n) \\ & \quad - Pr(A_1 \wedge A_2) - \dots - Pr(A_i \wedge A_j) - \dots \\ & \quad + Pr(A_1 \wedge A_2 \wedge A_3) + \dots + Pr(A_i \wedge A_j \wedge A_k) + \dots \\ & \quad \vdots \\ & \quad -(-1)^n Pr(A_1 \wedge A_2 \wedge \dots \wedge A_n). \end{aligned} \tag{2}$$

We define $p_i = Pr(A_i), \forall i \in \{1, \dots, a(l,x)\}, p_{ij} = Pr(A_i \wedge A_j), \forall i \in \{1, \dots, a(l,x) - 1\}, \forall j \in \{i + 1, \dots, a(l,x)\}, \dots, p_{1 \dots a(l,x)} = Pr(A_1 \wedge \dots \wedge A_{a(l,x)})$, and $S_1 = \sum_{i=1}^{a(l,x)} p_i$, $S_2 = \sum_{i=1,\dots,a(l,x)-1, j=i+1,\dots,a(l,x)} p_i p_j$, etc.

We first calculate $S_k$, the sum of the probabilities of all $k$ common intervals:

$$S_k = \binom{a(l,x)}{k} \frac{\prod_{i=1}^{P} \binom{a(l,x)-k}{f_i(l,x)-k}}{\prod_{i=1}^{P} \binom{a(l,x)}{f_i(l,x)}}.$$

From the above and from Equation 2 we can express the probability of at least one free interval in each attendees calendar under node $x$ as:

$$\sum_{j=1}^{a(l,x)} (-1)^{j+1} \binom{a(l,x)}{j} \frac{\prod_{i=1}^{P} \binom{a(l,x)-j}{f_i(l,x)-j}}{\prod_{i=1}^{P} \binom{a(l,x)}{f_i(l,x)}}. \tag{3}$$

Given the density profile characteristics of the attendee agents, we can use Equations 3 and 1 to calculate the probabilities of scheduling the meeting in any given number of iterations. Since we are considering finite length calendars, there may be a non-zero probability of failing to schedule the meeting. In that case, we have the probability of success for a finite number of iterations, and a probability of failure. As the probabilities of success over all iterations do not add up to 1, it is not a true probability mass function. We can normalize these probabilities, producing a probability mass function conditioned upon success of scheduling the meeting. Once the latter set of probabilities are calculated, we choose the search bias with smallest expected number of iterations to success.

### 4.3 Some examples

In the following, we will consider two different sets of DPCs (see Figure 1). Each set involve a host trying to schedule a meeting of length $l = 2$ hours with two invitees. All the agents are assumed to be managing calendars divided into 10 blocks of 5 hours each. We assume, in the case of the hierarchical search bias, that the temporal abstraction hierarchy is comprised of calendar hours, which are grouped into these blocks. The hierarchical negotiation mechanism, used by the host, first gathers information from the invitees about their respective calendar densities in each of the blocks, orders these blocks by the probability of successfully scheduling a meeting in each of the blocks, and negotiates over one block at a time going down the ordered list. As the first iteration involves exchanging information about the internal nodes of the abstraction hierarchy, meetings can be scheduled starting from the second iteration only.

Assuming meetings cannot straddle blocks, there are 4 intervals in each block that could have accommodated the meeting if the calendars were empty. So, the constant $K$ in equation 1, and terms $a(l,x), \forall x \in \{1, \dots, 10\}$ in equation 3, are equal to 4. The $n_x$ terms of equation 1, and $f_i(l,x)$ terms for all the attendees and for all the blocks in equation 3, are obtained from the DPCs of each agent.

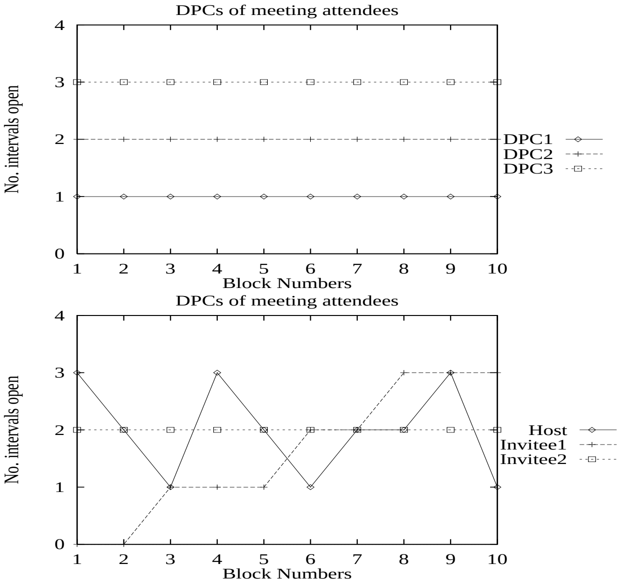


Figure 1: Some example density profile characteristics for host and invitee calendars (each calendar has 10 blocks of 5 hours each; desired meeting length is 2).

The first set of DPCs in Figure 1 corresponds to even density schedules for each of the agents, though the load on each agent is different. We will use this DPC to construct two examples, one in which the host has 1 interval open per block, and another in which the host has 3 intervals open per block. The second set of DPCs in Figure 1 presents a more complex scenario where the DPC curve has a constant value for only one of the three agents. We will use this set of DPCs to construct one example. We now present results from using the the hierarchical search bias and the linear early search bias with the best and good (proposing 3 intervals per iteration) announcement strategies on three example cases.

| Cases | **LE(best)** | **LE(good)** | **H** |
|---|---|---|---|
| **Case 1** | 2.575 | 1.294 | 3.575 |
| **Case 2** | 6.725 | 2.575 | 3.575 |
| **Case 3** | 10.68 | 3.836 | 2.193 |

Table 1: Expected number of iterations for scheduling the given meeting in three example cases using different search biases and announcement strategies.

**Case 1:** The host calendar corresponds to the curve DPC1 (one interval open per block). The host using the **LE** search bias with the best announcement strategy will end up announcing the open interval from successive blocks in successive iterations, taking at most 10 iterations within which it either schedules or fails at the end. The **LE** bias with the good announcement strategy proposes intervals in three different blocks per iteration, and hence requires a maximum of 4 iterations. The **H** search bias spends one iteration in the internal nodes of the abstraction hierarchy and then steps through the 10 blocks, one at a time, and hence requires a maximum of 11 iterations. The order of the performance of the different strategy combinations for this case is **LE** (Good) better than **LE** (best) better than **H**.

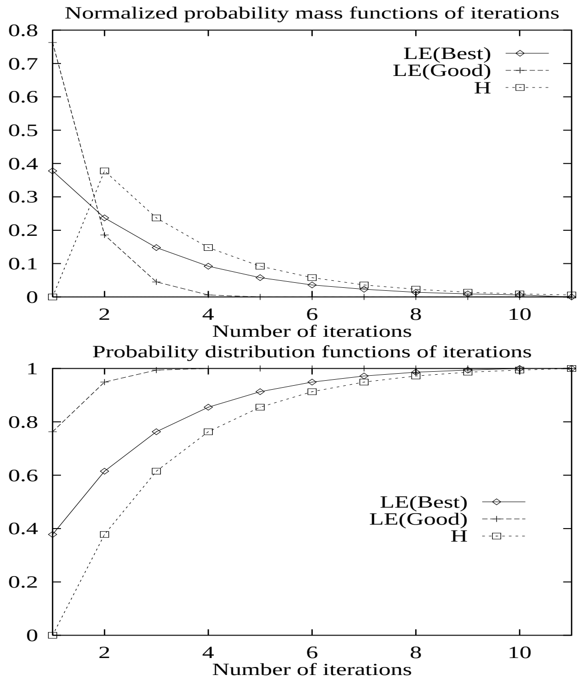


Figure 2: Normalized probability mass function and probability distribution function with different search biases for the first example.

**Case 2:** The host calendar corresponds to the curve DPC3 (three intervals open per block). The host using the **LE** search bias with the best announcement strategy will spend three iterations per block announcing the three open intervals in its calendar on any block in successive iterations. It will thus take at most 30 iterations within which it either schedules or will fail at the end. The **LE** bias with the good announcement strategy requires a maximum of 10 iterations, and the **H** search bias requires a maximum of 11 iterations. The order of the performance of the different strategy combinations for this case is **LE** (Good) better than **H** better than **LE** (best).

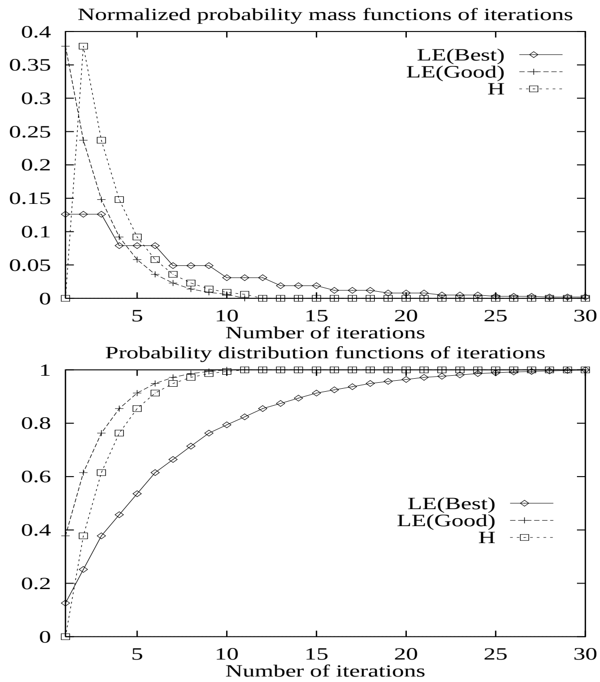


Figure 3: Normalized probability mass function and probability distribution function with different search biases for the second example.

**Case 3:** This case corresponds to the second set of DPCs in Figure 1. The host has a varying density schedule without any simple pattern inherent in it, one invitee has a front-loaded calendar (as would be produced by using a linear early type of search bias) and the other has an evenly loaded calendar (as would be produced by a hierarchical type of search bias). The maximum number of iterations taken by the **LE** search bias with best and good announcement strategies are 20 and 7 respectively, while the maximum number of iterations taken by the **H** search bias is 11. The order of the performance of the different strategy combinations for this case is **H** better than **LE** (Good) better than **LE** (best). In this scenario, the **LE** search biases waste iterations by proposing intervals open in the host on the first two blocks. The hierarchical search bias starts negotiation at the most empty part of the joint search space (corresponding to block number 9), and thus has a very high likelihood of scheduling the meeting in the second iteration (the first iteration is spent at the internal nodes of the abstraction hierarchy).

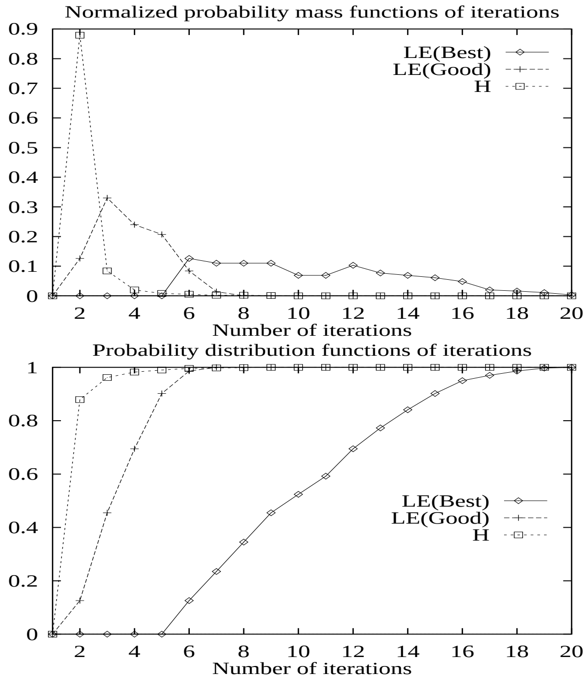


Figure 4: Normalized probability mass function and probability distribution function with different search biases for the third example.

The expected number of iterations taken by the search biases for each of the cases discussed above are listed in Table 1. The probability of failure to schedule the above meetings using any of the search biases is 0.009 for the first and second cases, and 0.004 for the third case. The probability mass functions conditioned upon successful scheduling and associated probability distribution functions for Cases 1, 2, and 3 are given in Figures 2, 3, and 4 respectively.

Let us now briefly discuss the qualitative characteristics of the DPCs of the host and invitees that makes it useful to use one search bias over the other. The hierarchical trades off iterations spent at inner nodes of the hierarchy with savings obtained by focusing on the relatively empty part of the joint search space. When compared to any other search bias on the metric of iterations required to schedule a meeting, the deciding factor becomes whether the savings obtained by focusing search to "greener pastures" can offset the initial cost incurred. When compared to the **LE** search bias, this means whether the host will spend sufficient unsuccessful iterations at the start of the calendar. This would happen if the host has a number of intervals free early in the calendar to propose that cannot be used because one or more invitees has its calendar relatively full in this part (as in **Case 2**). But the savings obtained by focusing search was not enough when compared to the **LE** search bias with the **good** announcement strategy. That finally happened in **Case 3**. More often than not, the hierarchical search bias will produce fewer expected iterations for cases where the densities of agent calendars vary and the different agent calendars look different (we believe these cases are more frequent in real life). Our analysis shows that it is better to choose search bias on a case by case basis rather than using a fixed option.

### 4.4 Interaction between strategies

The different strategy dimensions are not mutually independent. In order to use the hierarchical search bias, the corresponding informed negotiation mechanism has to be used by all meeting attendees. The different announcement and bidding strategy options can be used only with the linear early search bias. Different commitment strategies can be used with the hierarchical negotiation mechanism only when agents are exchanging information about the ground space of the calendar. As such, the sequence of strategy decisions to be made by meeting scheduling agents consist of choice of search bias followed by choices of announcement/bidding strategies, and ultimately the appropriate commitment strategy.

In order to develop the relative merits of different strategy combinations under different environments, we define the following characteristics of the environment and the agent schedules. In this paper, we will assume boolean values of high (H) and low (L) for these environmental factors (see Table 2).

**Success probability:** The probability that a proposed interval will be accepted by all attendees to a meeting will largely decide the options for various heuristic strategies. This probability can be calculated from the DPCs of attendees.

**Window of acceptance:** When a new meeting arrives, this factor is measured by the number of possible starting times for the meeting requested. As more and more intervals are unsuccessfully negotiated, the remaining window of acceptance reduces. We will use the dynamic window of acceptance measure to choose appropriate strategy options.

**Negotiation Density:** The number of meetings being currently negotiated in a given portion of the calendar impacts the likelihood of conflicts between processing of these meetings. This feature will be used to choose between different conflict avoidance options.

### 4.5 Announcement & Bidding Strategies

In this section, we assume that the scheduler has adopted the linear early search bias, and discuss the choice of the announcement and bidding strategies. If the number of iterations were the only metric we are interested in, we should almost always use the **good** announcement strategy. In practice, however, other performance metrics will be of importance as well, and we need to consider the effects of other environmental factors to decide on the most appropriate announcement strategy. After having chosen the linear early search bias, the scheduler knows the likelihood of success of scheduling using one iteration. Generally speaking, it is better to use the **good** announcement strategy whenever the negotiation density is low. The only exception is when the success probability and acceptance window are low as well. For this case, it is

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Success prob. | H | H | H | H | L | L | L | L |
| Acc. window | H | H | L | L | H | H | L | L |
| Neg. density | H | L | H | L | H | L | H | L |
| Announcement | B | G | B | G | B | G | G | G |
| Commitment | N | C | C | C | N | N | N | C |

Table 2: Matrix to choose the most appropriate announcement (G for **good**, B for **best**) and commitment strategy option (C for **committed**, N for **non-committed**) strategy options.

advisable to use **best** or **good** option depending on the relative priority of this meeting compared with other meetings being negotiated in the same region of the calendar.

The choice between alternative bidding strategies can be made based on the same factors as those used for making an informed announcement strategy choice. The scheduling agents should use an **alternatives** bidding strategy for conditions leading to a choice of **good** announcement strategy. Alternatively, the scheduling agents should use a **yes_no** bidding strategy for conditions leading to a choice of **best** announcement strategy. Another factor that determines expected iterations to convergence when an **LE** search bias is used, is the relative availability of the host with respect to the invitees at the front end of the window of acceptance of a given meeting. If the availability of the host is considerably higher than the most busy invitee, a large savings in iterations can be obtained when the corresponding invitee uses an **alternatives** announcement strategy compared to a **yes_no** announcement strategy. This helps the host to remove intervals from consideration, which it would have announced unsuccessfully otherwise.

### 4.6 Commitment Strategy

A scheduler decides on the appropriate commitment strategy after it has made the choice for search bias and announcement/bidding strategies. Table 2 shows that the deciding factor in the choice of commitment strategy is the success probability of scheduling the meeting using one proposal. An agent should commit to a proposed interval when the likelihood of scheduling the meeting with that proposal is high. The only exceptions are when

1. both window of acceptance and negotiation density are high (other meetings vying for the same spot, and there are enough other possibilities to schedule the current meeting)

2. both window of acceptance and negotiation density are low (not many possibilities remaining for this meeting, and few other meetings being considered for this part of the calendar).

## 5 Conclusion

We have demonstrated the benefits of adaptive choice of heuristic strategy combinations by automated meeting scheduling agents. Our proposed design for an adaptive meeting scheduling agent is based on expected performance of strategy combinations, developed using precise probabilistic analysis. The use of probabilistic analysis to develop quantitative estimates of the performance of alternative heuristic strategies is novel within the field of distributed AI (DAI). As such, our work proposes a new methodology to develop and evaluate future DAI systems.

Most distributed scheduling scenarios can be mapped into the distributed meeting scheduling framework. As such, our techniques can be used on a broader class of problems if the multi-stage negotiation protocol is applied to solve those problems.

We have not addressed the problems of constructing efficient user interface, calendar browsing software, encoding all possible user requirements, etc. These concerns need to be adequately resolved before a working application can be implemented. We plan to investigate the incorporation of user preferences within heuristic strategies as the next step towards the development of a functional meeting scheduling system.

## References


[1] K. Bogart. *Introductory Combinatorics*. Pitman, Boston, MA, 1983.

[2] S. E. Conry, R. A. Meyer, and V. R. Lesser. Multistage negotiation in distributed planning. In A. H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 367–384. Morgan Kaufman, 1988.

[3] S. Sen. *Predicting Tradeoffs in Contract-Based Distributed Scheduling*. PhD thesis, University of Michigan, October 1993.

[4] S. Sen and E. H. Durfee. A formal study of distributed meeting scheduling: Preliminary results. In *Proceedings of the ACM Conference on Organizational Computing Systems '91*, pages 55–68, November 1991.

[5] S. Sen and E. H. Durfee. A formal analysis of communication and commitment in distributed meeting scheduling. In *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence*, pages 333–342, February 1992.

[6] S. Sen and E. H. Durfee. The effects of search bias on flexibility in distributed scheduling. In *Working Papers of the 12th International Workshop on Distributed Artificial Intelligence*, pages 321–334, May 1993.

[7] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, Dec. 1980.

[8] E. Taub. Sharing schedules. *MacUser*, pages 155–162, July 1993.