# A Non-manipulable Meeting Scheduling System

Eithan Ephrati
Computer Science Department
University of Pittsburgh
tantush@cs.pitt.edu

Gilad Zlotkin
Center of Coordination Science
Sloan School of Management, MIT
gilad@mit.edu

Jeffrey S. Rosenschein
Computer Science Department
The Hebrew University
jeff@cs.huji.ac.il

**Abstract**

In this paper we present three scheduling mechanisms that are manipulation-proof for closed systems. The amount of information that each user must encode in the mechanism increases with the complexity of the mechanism. On the other hand, the more complex the mechanism is, the more it maintains the privacy of the users.

The first mechanism is a centralized, calendar-oriented one. It is the least computationally complex of the three, but does not maintain user privacy. The second is a distributed meeting-oriented mechanism that maintains user privacy, but at the cost of greater computational complexity. The third mechanism, while being the most complex, maintains user privacy (for the most part) and allows users to have the greatest influence on the resulting schedule.

## 1 Introduction

The basic research problem in meeting scheduling is that of timing, that is, when to set a meeting. This question becomes more complicated when there are several meetings to be scheduled that involve the same resources (mainly participants), and as the number of participants (constraints) grows [14, 18]. One key question is how to allocate all the feasible time slots for a meeting [20, 21]. Another key question is how to choose, among these possible time slots, the most appropriate one for a particular meeting [26]. This paper is concerned with both of the above questions.

We distinguish between two paradigms of meeting scheduling scenarios: *Open Scheduling Systems* and *Closed Scheduling Systems*. In Open Systems, one or more *independent* autonomous individuals try to schedule a meeting. These individuals are completely in control of their own time resources, and have no obligation to meet one another (unless that serves their own selfish interests). Thus, the individuals themselves determine the feasibility of a meeting.

On the other hand, in Closed Systems, such as a business or an organization, the meeting mechanism is *imposed* on the members of the system. The participants of any potential meeting belong to the same body. They thus have some obligation to take part in a meeting, if feasible. The constraints regarding the meeting are determined by the system and not by the participants. Each member of the organization (e.g., employee) is assumed to meet with others as much as needed. In some sense, the organization has a degree of ownership over the user's time. A closed scheduling system also maintains a consistent and complete global calendar of the organization's members.

In both kind of systems, we would want the schedule of meetings to be optimal in some sense. The optimality of a meeting's timing should be determined, as much as possible, by the potential individual participants based on their individual preferences. If these preferences could be known perfectly by the system, scheduling becomes an optimization problem. However, since this is usually not the case, part of the scheduling problem is to extract from the users their true preferences regarding potential meeting times.

There are several reasons that Closed Systems are more easily dealt with than Open Systems. First, we will be using an economics-based point system to allow users to specify their preferences; in a Closed System, we can easily control the number of points that users get, and reasonably assume that points have roughly the same meaning for different users. At times, we will need to be able to "collect" these points by levying a kind of tax; this is more easily implemented in a Closed System. Also, by keeping track of how many points users have, we can make sure there is no side-trading in points among users, which would undermine our mechanism. A Closed System can monitor which meeting times are available and which are not; we make use of this capability, which is harder to ensure for an Open System. And finally, in an Open System, a user might choose to avoid a meeting entirely, while in a Closed System (as mentioned above), we assume that agents have a responsibility to attend meetings—we simply want to schedule them in some optimal way.

In this paper we introduce a meeting scheduling algorithm for Closed Systems. Our system has several desirable attributes, of which the most important is non-manipulability. We consider three alternative approaches to the scheduling process.

To use a Closed System scheduling mechanism, a user must subscribe to the system, and the mechanism cannot be used to schedule meetings with other users that are not subscribed. Meeting scheduling beyond the boundaries of a single closed system may require interaction between two or more closed scheduling systems, and is beyond the scope of this paper.

## 2 Group Decision Mechanisms

In its general outlines, the problem of meeting scheduling is a typical problem of group decision. The issue of group decision has been a major research agenda in several research fields, including game theory, economics, social welfare theory and voting theory. The main question that this research addresses is, given the individual (private) preferences over alternative mutual decisions, what should the mutual decision be. As is the case with the systems that we propose, researchers in these fields have been looking into mechanisms that will allow the individuals to converge (agree) on one global decision. And just as we aspire for the system's schedule to be "optimal" in some sense, they have been defining the desirable attributes of the decision process and looking for procedures that have these attributes.

Different kinds of desirable attributes of decision functions that characterize optimality have been suggested in game theory, economics, and voting theory. Typically, the attributes are concerned with the influence of an individual user on the outcome, and the impact of the outcome on the individual. Some common criteria include Pareto Optimality, Fairness, and Individual Rationality.[1] In this section we briefly summarize the most common criteria.

[1] The issues of solution criteria are discussed in [1], in the context of combining various default theories.

## 2.1 Attributes of Social Decision Processes

In general, the optimality of the decision process may be viewed with respect to two main aspects/categories: attributes of the resulting decision, and attributes of the group decision mechanism itself. Below, we mention some of the more common criteria that have been addressed in the literature [19, 27, 13, 11]:

**Attributes of the resulting decision:**

- Global optimality — The chosen alternative should be optimal in some global sense. The most common requirement is that the decision will be Pareto Optimal, meaning that it is impossible to change the decision in a way that will make some users better off without making some other users worse off. This same attribute is sometimes called Unanimity, to denote that if alternative $X$ is preferred over some other alternative $Y$ by all users (unanimously) then $Y$ should not be chosen. Within the "pareto frontier" (all the Pareto Optimal decisions), there are many additional criteria of optimality based on Social Welfare theory. We discuss this issue further in Section 2.2.
- Condorcet Winner — The chosen alternative should beat any other alternative in a pairwise contest.
- Nash's Independence of Irrelevant Alternatives — If out of a set of alternatives, $X$ is chosen, then if any other alternative $Y$ is removed from the set, $X$ will still be chosen. In other words, the choice should remain unchanged when the group is presented with a subset of the original group that includes the original choice.
- Monotonicity — If $X$ is to be chosen by the process and one or more users change their preferences *in favor* of $X$ (without changing the order of preferences over the other alternatives), then $X$ should remain the choice.

**Attributes of the process itself:**

- Individual Rationality — A user may only gain (utility) by taking part in the process (compared to not participating).
- Simplicity — The process should be simple in two respects: the computational complexity of determining the choice, and the individual computational complexity of determining each user's own behavior (strategy) in light of the rules of the process.
- Stability — The behavior of the participants that would lead to the desired outcome should (under some assumptions) be their optimal one (i.e., a [rational] user would not be tempted to deviate from the desired behavior that we want the process to induce). Usually this behavior (strategy) should be in some equilibrium (in Section 5.1 we describe the main notions of equilibrium from Game Theory).
- Privacy Preserving ("Information Decentralization") — The behavior of each individual should depend on as little information as possible regarding the others (preferences and behavior), and the choice function should depend on as much of a global view as possible (in

contrast to taking into account interactions among individual preferences). For example, it is considered preferable if one's behavior can be determined according to others' aggregated behavior instead of having to take into account the individual behavior of each other member.

- Decentralization — The degree of distribution affects the likelihood of a bottleneck, the fragility of the process, and the need for a central decision maker.

- Expressiveness — Most decision mechanisms consider only the Ordinal preferences of the users. The magnitude by which some alternative is preferred to another cannot be expressed. However, there are mechanisms that allow more powerful rating of preferences, such as rating the alternatives by points, or the assignment of actual (cardinal) utilities to preferences. The more expressive the rating is, the more informed can be the choice that is made.

- Symmetry — Given the possible permutations of the users' roles in the process, the outcome should remain the same regardless of these permutations. The strongest kind of symmetry is *Anonymity*, which says that the process answers all possible symmetries (the identity of a user has absolutely no influence on the outcome).

### 2.2 Optimality and Social Welfare

The most important attribute of the resulting decision is optimality. In our scenario, optimality is measured by the global convenience of the users. However, there are many ways to measure global convenience, and it is not obvious how the system designers will decide on one or another. Considerations other than pure convenience values (such as priority and fairness) might need to be taken into account. For example, it might be desirable in some scenarios to look for a schedule that maximizes the median of the stated conveniences or some weighted sum of these statements, or, following an egalitarian approach [23] we might want to maximize the minimal stated convenience ($\max\min_i c_i$) or minimize the differences in convenience ($\min\max_{i,j} \mid c_i - c_j \mid$).

One simple common approach (due to Nash [16, 17]) is to choose the outcome that maximizes the product of the individual conveniences ($\max\prod_i c_i$). This approach guarantees a relatively fair distribution of the mutually earned convenience, but narrows the space of feasible consensus states. A negotiation protocol for autonomous users that follows this approach may be found in [30].

In this research, however, we take the viewpoint of the system designer. We therefore follow the pure utilitarian approach, and prefer consensus group decisions that maximize the sum of the individual users' utilities. In contrast to the approach that maximizes the product, we would rather have, for example, a schedule that gives two users a convenience of 0 and 11 respectively, over a schedule that gives each of them a convenience of 5. A further discussion of this approach may be found in [10].

## 3 Monetary-Based Meeting Scheduling Systems

The underlying idea of our scheduling system is economic in flavor. We create a market of "convenience points," that will allow users to efficiently reach agreement on meeting times. The system simulates a closed primitive economic market. Convenience points are used to express user preferences over alternative schedules. The system keeps track of the "convenience balance" that each user has. If desired, points may be consumed by the system or distributed to the users; however, users may not trade points among themselves directly.

There are two kinds of interactions that a user can have with the scheduling system:

1. Initiate a meeting $\mathcal{M}_i^t = \{u_1, \ldots, u_m\}$ — At time $t$ user $i$ requests the setting of a meeting, by specifying the parties to be involved $(u_1, \ldots, u_m)$.

   It is up to the system administrator to decide which of the users are authorized to initiate a meeting. We here assume that any user is authorized. If it is desired, the initiator may also specify the time windows, $tw(\mathcal{M})$, in which the meeting may take place.

2. Express convenience — A user expresses his preferences over possible meeting schedules. These preferences are expressed by assigning convenience points to different schedules.[2]

Below, we describe three approaches by which the system may identify schedules that approximate the maximal global convenience. We first describe the naive algorithm that corresponds to each approach. In Section 5 we describe how these algorithms can be made more powerful so as to deal with more sophisticated (or manipulative) users.

The three algorithms differ in complexity, the amount of influence that the user has on the final schedule, and the kind of information that the user has to reveal about his individual preferences. In general, the less complex the process is, the less influence the user has on the details of the schedule, and the more privacy the user attains, the more complex the algorithm becomes.

### 3.1 Calendar Oriented Scheduling

In a calendar oriented closed scheduling system, each user expresses his preferences over all the available time slots in his calendar. These preferences are *only* over time slots, regardless of the actual meetings to be assigned to the time slots, and are declared only once for the entire time period considered by the system. The global calendar is simply an aggregation of all individual users' calendars (with their expressed preferences). The calendar includes all feasible time slots (i.e., all time slots that as far as the system is concerned *could* be scheduled for meetings).

The "time horizon" of the system is determined by the latest meeting that can be set. To represent his preferences, each user spreads convenience points over all time slots between the present/current time and the time horizon. When a meeting, $\mathcal{M}$, is initiated, the system inspects all the present profiles of the participants and identifies which is the most convenient time slot for that meeting. This is exactly the slot in which the sum of stated convenience points is maximal. That time slot is labeled with the meeting and becomes infeasible for any other meeting that involves any $u_i \in \mathcal{M}$.

More formally, we have the following assumptions and definitions:

- The calendar is divided into $[t_1, \ldots, t_n]$ basic time slots (e.g., 60 minutes each).

[2] Since preferences grade alternatives with respect to one another, expressing preferences could most naturally be done by using a graphical interface (e.g., [?]). Note that if a user does not wish to accurately calculate the number of convenience points for each alternative, he may use his points to express his *general* preferences over alternatives (i.e., if alternative B is preferred to A, just rank B higher than A regardless of "how much" B is preferred). This kind of less expressive preference assignment is used by most existing meeting schedulers (e.g., [?]).

- At any time, the system has some future time slot such that beyond that time slot no meeting can be set. This is $T^t$, the "horizon" of the system at time $t$ (e.g., it can be updated at the end of every week to include the following two weeks). The value of $T$ may be determined by the system's administrator (independently of the users' requests). Or, in the case where a meeting initiative includes the desired time slots, the horizon may be set dynamically to be the furthest initiated time slot.

- Each new user, $i$, introduced to the system is presented with $q_i^0$ convenience points (as determined by the system administrator).

  For each time-slot $t$, between the current time, $t^*$, and the time horizon of the system $T^{t^*}$, each user $i$ is given an additional quota of convenience points, $q_i^t$ (as determined by the system administrator).

  Each user may spread convenience points over the calendar. $c_i^t$ denotes the number of convenience points that $i$ assigns to time slot $t$.

- $w_i^{t^*}$ denotes the wealth of user $i$ at time $t^*$ (that is, the sum of all his given convenience points, minus his points that were consumed by the system).

  At all times the number of convenience points that each user may assign to any slot may not exceed its wealth at that time ($(\max_{t=t^*}^{T} C_i^t) \leq w_i^{t^*}$). That is, the wealth of a user determines the upper bound of convenience points that he can assign to *any particular time slot*. The total of the convenience declarations for more than one slot may exceed the wealth bound.

In the naive system, all users are given the initial quota at the initialization of the system. No points are consumed by the system and no other points are given. Then, whenever a new horizon is set, each user is required to declare his convenience, $c_i^t$, for each time slot. The only restriction is that the number of points, per slot, will not exceed his wealth at the time.

Essentially, the scheduling algorithm is a greedy algorithm—the Calendar Oriented Scheduling system executes a simple loop:

1. Whenever a meeting, $\mathcal{M}$, is initiated, the system finds the set, $\mathcal{T}$ of time slots that are available/feasible for that meeting.[3]

2. The system allocates $\hat{t}$, which is the maximizer of $\max_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} c_i^t$. When this slot is identified, the meeting is set and the slot becomes blocked for any further meeting with any $u_i \in \mathcal{M}$. (Note that the users are not required to redeclare their preferences.)

The complexity of matching a meeting with a time slot is simply linear in the time horizon (actually, in the feasible fraction of that horizon). However, this scheduling algorithm does not guarantee the ultimate optimal schedule. It might be the case that to derive the actual convenience maximizing schedule, some meeting should be rescheduled due to the arrival of new meetings.
**Example:** Assume that are only two available time slots $\langle t_1, t_2 \rangle$ and three users $(u_1, u_2, u_3)$ with the following declared preferences respectively: $\langle 11, 0 \rangle, \langle 0, 0 \rangle$, and $\langle 33, 0 \rangle$ (i.e., $u_1$ prefers $t_1$ over $t_2$

[3] In the general case, these slots would simply be the conjunction of the individual available/feasible slots. However the system may incorporate other constraints into the decision, such as available meeting rooms, and further narrow the feasible time space (see Section 8). If the initiator of a meeting is allowed to specify a time window for the meeting (i.e., $tw(\mathcal{M})$), then the feasible space will be further pruned.

by 11, $u_2$ is indifferent, and $u_3$ also strongly prefers $t_1$). Now assume that $u_1$ initiates the meeting $\{u_1, u_2\}$. The sytem will choose time slot $t_1$ with a score of 11 (if the meeting had been set to $t_2$ it would have scored 0). Later on, $u_3$ initiates the meeting $\{u_1, u_2, u_3\}$, which if set to $t_1$ would score 44; since this is not feasible, the meeting is set to $t_2$ with a score of 0.

This example demonstrates the weakness of the greedy approach. As is the case with the following algorithms, the schedule may become more efficient in exchange for more complexity. For example, the matching of meetings to time slots may be done only after several meetings, say $m$, have been initiated. Then the system should solve the following maximization problem (for each $m$ meetings) $\max_{t,j} \sum_{i \in \mathcal{M}_l} c_i^t$. In the example above, $m = 2$ would yield the more efficient schedule.

The main advantage of the Calendar Oriented approach is that it is relatively simple to calculate the schedule. But this simplicity is due to its main drawback: all meetings are handled symmetrically. Thus, the user cannot rate his preferences with respect to a specific appointment. The fact that the user can say only *when*, and not *what*, confers a lot of power to the system.

This attribute may not always be desirable—under many circumstances, the nature of a meeting may influence users' preferences (e.g., if Jeff is weighing a meeting with Einstein vs. a meeting with Zlotkin, time might be the last thing about which he'd be concerned).

### 3.2 Meetings Oriented Scheduling

To enable users to embed in their preference profile their evaluation of the *nature* of the meeting as well, we use a variant of the previous approach. According to this "Meetings Oriented" approach, preferences are expressed per alternative specific schedule (rather than over the global time space). Users assign convenience points to combinations of meetings and time slots. Convenience points are given per meeting rather than per time-frame.

More formally, we have the additional following assumptions and definitions:

- At the initialization of the system, each user $i$ is given $q_i^0$ convenience points.
- As each meeting $\mathcal{M}$ enters the system (as a feasible request), each user $u_i \in \mathcal{M}$ is given an additional quota of convenience points, $q_i^{\mathcal{M}}$.

The naive algorithm that follows this approach is similar to the previous one. Each user is given a constant number of convenience points. The scheduler operates as a FIFO queue; each meeting is handled in the order of its initialization. When a meeting arrives, all the parties involved declare their available time slots for that meeting (if desired by the system's administrator, otherwise all feasible slots are considered available). Then $\mathcal{T}$, the intersection of all available slots, is computed. Finally, each $u_i \in \mathcal{M}$ assigns convenience points $c_i^t$ to each time slot $t \in \mathcal{T}$. Thus in this approach the available time slots are set dynamically, at the initiation of a meeting; in the calendar oriented scheduling system, the available slots were set prior to the initiation of any meeting.

As before, the slot that maximizes $\sum_i c_i^t$ is identified and labeled with the meeting; that time slot is now unavailable for any further meetings that would involve any of the participants.

Thus, the preferences over time may be changed dynamically by the users, with respect to the content of the meeting involved. However the complexity of the process increases significantly; since for each appointment the entire time horizon is reevaluated, the complexity becomes the order of the number of time-slots multiplied by the number of appointments. However, this mechanism is more informative for the users and therefore able to increase the optimality of the schedule. (As in the previous case, if users are allowed to state the potential time windows for initiated meetings, the feasible time space will be pruned.)

**Example:** Three users, $u_1, u_2$ and $u_3$, subscribe to the system. The time slots are the morning, mid-day, and evening. $u_1$ has 5 points of wealth and prefers the evening. $u_2$, with a wealth of 4, does not like to meet in the morning, and is indifferent between mid-day and evening. $u_2$, however, suffers from nightmares if he happens to see $u_1$ in the evening. $u_3$, with wealth of 7, prefers mornings over mid-days and mid-days over evenings. In calendar oriented scheduling, the users need to state their calendar regardless of the meeting involved. Thus, the calendar would look like Figure 1.

Now, assume that all three users need to schedule a meeting for tomorrow. The system will schedule the meeting to be in the evening.

| | **Morning** | **Mid-Day** | **Evening** |
|---|---|---|---|
| $u_1$ | 0 | 0 | 5 |
| $u_2$ | 0 | 4 | 4 |
| $u_3$ | 7 | 4 | 0 |
| Sum | 7 | 8 | 9 |

Figure 1: Calendar Oriented Scheduling

In meeting oriented scheduling, on the contrary, the users will be asked to state their preferences about the specific meeting. Thus $u_2$'s preferences will be changed. The resulting rating can be seen in Figure 2.

| | **Morning** | **Mid-Day** | **Evening** |
|---|---|---|---|
| $u_1$ | 0 | 0 | 5 |
| $u_2$ | 0 | 4 | 0 |
| $u_3$ | 7 | 4 | 0 |
| Sum | 7 | 8 | 5 |

Figure 2: Meeting Oriented Scheduling

In this case, the meeting will be scheduled to be at mid-day. As in the previous case, the meetings are being dealt with one at a time (in the order in which they are initiated); thus, this algorithm may result in a sub-optimal schedule. The next technique we examine, Schedule Oriented Scheduling, aims at finding the *actual* optimal schedule.

### 3.3 Schedule Oriented Scheduling

In Schedule Oriented Scheduling all possible *schedules* are taken into consideration. Instead of expressing their convenience with the time set for specific meetings, users express their preferences over entire schedules. Each schedule identifies one possible ordering of future meetings.

Similar to the previous approaches, we have the following additional assumptions and definitions:

- As before, the system's calendar is divided into time slots. Each $k$ slots constitute one time frame (e.g., 8 hours is a full working day). There are $[f_1, \ldots, f_{\lceil \frac{n}{k} \rceil}]$ frames.
- At the initialization of the system, each user $i$ is given $q_i^0$ convenience points.
- As each meeting $\mathcal{M}$ enters the system (as a feasible request), each user $u_i$ of the system is given an additional quota of convenience points, $q_i^{\mathcal{M}}$.
- $\mathcal{S} = \cup_j S_j$ is the set of feasible schedules. Each schedule is one feasible match of all the meetings that have been initiated up to the present time, $p$, with time slots in the horizon of the system.[4]

  $c_i(S_j)$ is the number of convenience points that $i$ assigns to schedule $S_j$. As in the two other procedures, this number cannot exceed $i$'s wealth.

In this case, the naive algorithm would, iteratively, find all possible schedules (from the present time to the horizon—the further the horizon is, the more accurate the result is), and have the users express their preferences over the possible schedules. Then, the first time frame of the desired schedule is to be "frozen" and followed. (Meetings that were not matched within these frames will be considered again in the next iteration.) When the time frame ends, the process is repeated (with all new and old unmatched meetings)

This process best approximates the optimal schedule (if only the next time slot is frozen, and the horizon is infinite, the schedule is optimal). However the complexity is very high; it is in the order of the factorial of the number of meetings multiplied by the number of time-slots (p $! \times \mid T \mid$). The administrator of the system may choose to reduce this complexity by setting a closer horizon, further constraining the feasibility of meetings, or enlarging the time frames that should be frozen. Another way to reduce complexity (at the expense of optimality) is to use any mixture of this approach with the previous one (e.g., determine a [fixed] schedule for each $k$ meeting initiatives that enter the system).

# 4 Beneficial Manipulation in Meeting Scheduling

The above mechanisms have many desirable attributes: the choice approximates the optimal utilitarian choice, it is Pareto Optimal, and it chooses the Condorcet Winner. The process itself is symmetric, relatively simple, satisfies monotonicity, is based on cardinal preferences, and maintains some privacy. However, the three mechanisms all suffer from a severe drawback: they are all sensitive to manipulation.

The individually motivated user has an incentive to leave open only the time slots that are most convenient to him. He may be motivated to block all other time slots that are slightly less convenient to him.

The underlying issue has to do with concessions. A user, by keeping only his most convenient time slots open, is basically counting on the other users to be flexible, and meet when it suits him. Of course, when many users follow this line of reasoning, they are all motivated to limit their own flexibility, leading to difficult (if not impossible) scheduling. This scheduling problem is thus

[4] As before, the feasibility of a match may be further constrained by the time windows specified by individuals.

a classic instance of a multi-agent game of chicken [22]—each user wants to be inflexible himself, counting on other users to show flexibility.

The prototypical deception in such meeting scheduling mechanisms is to undervalue, or simply block, time slots such that the meeting will be forced to be at your most convenient time slots.

**Example:** Consider again the previous example from Figure 2. Meeting oriented scheduling chooses mid-day, because both $u_2$ and $u_3$ would rather have the meeting at mid-day and not in the evening. But by being dishonest, $u_3$ (who prefers mornings) can do better. He should simply declare $\langle 7, 0, 0\rangle$ (instead of his true preferences, $\langle 7, 4, 0\rangle$), and make morning be the group choice. This phenomena is even more likely if we allow users to state their available time slots for a meeting: a user may declare that the available slots are only the ones that he most prefers (e.g., $u_1$ would state that the morning is out of the question).

We thus seek a process that will stabilize the system, by making true declaration of convenience the most beneficial behavior that a user can adopt. This highly desirable attribute has not been considered in previous meeting schedulers (see for example [25, 26]).

# 5 Stabilizing the System

In this section we show how the scheduling algorithms that were presented above can be made resistant to manipulation. We show how to embed within each of the procedures a mechanism, the Clarke Tax, that will guarantee the honesty of the users.

## 5.1 Game Theoretic Concepts of Solution

Game theory has addressed many interactions similar to the one considered here. Such interactions have been analyzed so as to determine what a user's chosen strategies would be, given the rules of the interaction. Our aim is complementary; it is to *design* rules that would induce the users to adopt some specific strategy that we consider to be desirable.

To be motivated to adopt a particular strategy, a rational selfish user should be convinced that that strategy is superior in some sense to his other alternative strategies. The most common solution in game theory derives cooperation as the best response to the other users' cooperative behavior:

**Definition 1** *The strategy combination is a* Nash equilibrium *if no user has an incentive to deviate from his strategy given that the other users do not deviate.*

This concept of solution was used for example (within the Distributed Artificial Intelligence literature) in [31, 32]. The drawback of the Nash equilibrium is that in general there are several equilibrium points for the same game, and the desirability of a strategy is considered only from a player's viewpoint at the start of the game (not taking into consideration *all* possible paths of the game). Thus, it might be difficult to have the group of users converge to a specific equilibrium and the equilibrium point may be sensitive to the dynamics of the interaction (see, however, Section 6).

A stronger concept of solution (the strongest concept within the hierarchy of equilibrium points), is to motivate the user to follow the desirable behavior regardless of what the others do. Such motivation would be accomplished if that strategy would be proven to be (under the rules of encounter) the best one given *any* strategy of the other users.

**Definition 2** *The strategy is a* dominant strategy *if it is a user's strictly best response to any strategies that the other players might pick, in the sense that whatever strategies they pick, his payoff is highest with his dominant strategy. A* dominant strategy equilibrium *is a strategy combination of each player's dominant strategy.*

Thus, in our scenario, the most attractive solution would be to have some particular behavior with certain desirable properties as each user's dominant strategy. Having a mechanism that induces an equilibrium point which is the result of a dominant strategy is very desirable from the viewpoint of computer interaction. The fact that there is no importance to the other users' behavior does away with the need to reason about the other users' strategies, knowledge, or even computational capabilities. The behavior of a user depends solely on its own characteristics. Thus, in comparison to other approaches, the individual complexity of decision-making is reduced significantly. This concept of solution was used in [4, 2, 3].

In the sections below, we focus on the design of systems that induce a solution in dominant strategy equilibrium. In Section 6 we discuss how users may maintain more privacy about their private preferences, if the system designer would be satisfied with the somewhat weaker concept of Nash strategy equilibrium.

### 5.2 The Clarke Tax Mechanism

The Clark Tax Mechanism (CTm) is one of many one-shot voting-by-bid mechanisms that were invented within the fields of voting theory and economics. Unlike other mechanisms, the CTm is non-manipulable; the basic idea of Clarke's Mechanism is to make sure that each voter has only one dominant strategy, telling the truth. This phenomenon is established by slightly changing the classic sealed-bid mechanism: instead of simply collecting the bid of the winning bidder, each user is fined with a tax. The tax equals the portion of his bid that made a difference to the outcome. The example in Figure 3 shows how to calculate this tax. Each row of the table shows several pieces of information regarding a user. First, his preferences for each schedule are listed. Then, the total score that each schedule would have gotten, had the user *not* voted, are listed. An asterisk marks the winning choice in each situation.

| | True worth of each schedule | | | Sum for each schedule without $i$ | | | Tax for $i$ |
|---|---|---|---|---|---|---|---|
| | $s_1$ | $s_2$ | $s_3$ | $s_1$ | $s_2$ | $s_3$ | |
| $u_1$ | 63 | 3 | 42 | 98 | *167 | *167 | 0 |
| $u_2$ | 0 | 48 | 60 | *161 | 122 | 149 | 12 |
| $u_3$ | 27 | 60 | 21 | 134 | 110 | *188 | 0 |
| $u_4$ | 18 | 21 | 69 | 143 | *149 | 140 | 9 |
| $u_5$ | 53 | 38 | 17 | 108 | 132 | *192 | 0 |
| Sum | 161 | 170 | *209 | | | | |

Figure 3: Calculating the Clarke Tax

For example, when all the users voted, schedule $s_3$ was chosen. If $u_2$ had not voted, $s_1$ would have been chosen. The score in this situation would have been $\langle 161, 122, 149 \rangle$, and $s_1$ would have beaten $s_3$ by 12. Thus, user $u_2$ has affected the outcome by his vote, and he has affected it by a "magnitude" of 12; he is therefore fined 12. Users $u_1$, $u_3$, and $u_5$ are not fined because even without the vote of each of them (separately), $s_3$ would still have been chosen.

Given this scheme, revealing true preferences is the dominant strategy. A user that overbids (so that some given schedule will win) risks having to pay a tax larger than his true preferences warrant. Similarly, the only way to pay less tax is to actually change the outcome—and any user that underbids (to change the outcome and save himself some tax) will always come out behind; the saved tax will never compensate him for his lost utility.

The Clarke Tax decision mechanism is appealing for several reasons. First, it is not manipulable by individuals—any other declaration of preferences is *dominated* by declaring the truth. Therefore, it saves each user from the computational complexity of guessing what the others' preferences and strategies are, what the negotiation set is, and how it can be manipulated. This simplicity of strategy is highly desirable in the design of automated agents. Agents tell the truth out of their own self-interest—there is no need to assume that users will act benevolently by design. Thus, the process answers both the "simplicity" and "stability" criteria.

A second advantage of the technique is that it satisfies several desirable criteria, including the "Condorcet Winner" (—a choice that would have beaten every other choice in pair-wise votes is guaranteed to be chosen by the mechanism [6]), "monotonicity" (—by giving an alternative a higher value, a user cannot undermine the alternative's selection), "independence of irrelevant alternatives" (—removal of any "unchosen" alternative from the set of alternatives will not change the outcome [28]), "individual rationality" (a user may only gain utility by taking part in the process), "anonymity" and "neutrality" (—the identity of a voter or the name of an alternative has no influence on the outcome), "expressiveness" (—preferences are expressed using the actual cardinal utilities), it is relatively simple, and finally the process can be designed to preserve privacy since the actual choice function uses only the total sum of preferences. In previous work we have also shown how the mechanism may be distributed [5] and administered in a way that maintains even more privacy [4].

A third advantage is that the alternative chosen by the Clarke Tax mechanism answers a social welfare criterion similar to the summation criterion mentioned above [29]. In fact, the Clarke mechanism is just one member of the family of Groves mechanisms [9]. It has been proven in Economics that any decision mechanism that chooses a schedule with the same properties as the CTm does, and that also has telling the truth as a dominant strategy, belongs to this family [8]. However, the CTm requires the least amount of tax to be paid, from among the members of this decision mechanism family [12]. It guarantees the best minimal utility level for each of the participants, and is the only mechanism within this family that has no free-rider problem [15] (i.e., a user will not be tempted to avoid the process, hoping to benefit from the decision without the risk of paying the tax).

All these good attributes make the mechanism a very desirable tool for meeting scheduling. Fortunately, the mechanism can be employed in the scheduling system that we have presented above, in a relatively direct manner.

## 5.3 Employing the CTm in our Systems

To embed the CTm within the scheduling techniques that we suggested above, the three scheduling procedures and the CTm itself should be slightly changed. The underlying idea is that users will be taxed according to the CTm, at each time a scheduling decision is being made. Taxation is done by having the system consume the tax-corresponding number of convenience points. (As mentioned above, this is one of the main reasons we need the system to be a Closed System.) However, since points are consumed by the system, points should also be distributed. A critical aspect of the CTm, is that the "income" of a user must be *independent* of the decision process. Therefore, points are distributed equally and periodically. Somewhat more formally, the previous techniques should be updated as follows:

1. Users should be taxed (according to CTm procedures) whenever a schedule is determined. (In the first two techniques this should be per meeting and in the third per schedule.)

   The wealth, $w_i^t$, of each user is updated accordingly so as to ensure that in consecutive schedulings its statement of preferences will not exceed this wealth.[5]

2. Each user should be given additional convenience points periodically, $q_i^t$, so as to prevent bankruptcy. Under the first technique above, this should be done after each $k$ time slots that enter the horizon; under the second technique, after $k$ new meetings have entered the system; and under the third technique, after $k$ time frames have been scheduled.

The essential idea is that the more points a user has, the greater is the difference between alternatives he can declare, the more expressively his preferences can be stated, and thus the more he influences the schedule. Thus each user is motivated to be taxed as little as possible. However, to avoid taxes the user has to concede as much as possible. A user that does not concede will run short of points, and that may happen exactly when these points are really needed.

To make the process completely non-manipulable we cannot allow the users to have influence on the feasibility of time slots. Therefore, the updated mechanism cannot allow any specific time window to be attached to an initiated meeting (otherwise a user might try to exclude even slightly inconvenient time slots).

The original Clarke Tax mechanism should also be slightly changed; we use the following definitions to specify the schedule oriented CTm:

- The function $c_i : \mathcal{S} \rightarrow \mathcal{C}$, returns the *true* convenience (to $u_i$) of each schedule.[6] The function $d_i^k(j)$ returns the *declared* convenience of schedule $s_j$ by user $u_i$ at scheduling step $k$. $\vec{d_i^k}$ denotes the vector $\langle d_i^k(1), d_i^k(2), \ldots, d_i^k(m) \rangle$, the user's declared convenience over all $m$ alternative schedules.

- The profile of preferences declared by each of the $n$ users at step $k$ is denoted by $D_n^k$, where $D_{\neg i}^k$ denotes this set excluding $i$'s preferences, such that $D_n^k = (D_{\neg i}^k, \vec{d_i^k})$.

[5] Notice that due to this restriction the first method may become more complex. The initial assignment of convenience points of some users may not be valid after they have been taxed; these users must then reassign their remaining points.

[6] A schedule refers to a different thing in each scheduling method: in the first, a schedule is just a time slot; in the second, it is a match of a meeting and a time slot; and in the third, it is a full schedule.

- The choice function $f : D_n^k \times \mathcal{S} \rightarrow \mathcal{S}$ returns the schedule that is the *maximizer* of $\sum_{i=1}^{n} d_i^k(s^k)$.
- The tax imposed on $i$ at step $k$ is $t_i^k(f(D_n^k)) = \sum_{j \neq i} d_j^k(f(D_{\neg i}^k)) - \sum_{j \neq i} d_j^k(f(D_{\neg i}^k, \vec{d_i^k}))$, if this value is positive. Otherwise, $t_i^k$ will be zero. Therefore, the benefit $b_i^k(f(D_n^k))$ of user $i$ with respect to the chosen alternative is $c_i(f(D_n^k)) - t_i^k(f(D_n^k))$ (the convenience it associates with the decision, minus the tax consumed given that decision).

**Example:** Consider again the example from Figure 2. Using the CT mechanism the meeting will still be scheduled at mid-day, but now the system will tax $u_2$ by 3 points and $u_3$ by 1 point (as can be seen in Figure 4). $u_1$ will not be taxed.

| | **Morning** | **Mid-day** | **Evening** | **Tax** |
|---|---|---|---|---|
| $u_1$ | 0 | 0 | 5 | 0 |
| $u_2$ | 0 | 4 | 0 | 3 |
| $u_3$ | 7 | 4 | 0 | 1 |
| Sum | 7 | 8 | 5 | |

Figure 4: Meeting Oriented Scheduling with CTm

$u_2$ gained utility of 1, since he rated the value of the mid-day meeting with 4 and paid only a tax of 3. Similarly, $u_3$ has gained utility of 3. $u_1$ got utility of 0 since he rated a mid-day meeting with 0 value. However, since he was not taxed he can use his 5 point-bound to be more decisive in future scheduling, while in future steps $u_2$ and $u_3$ will be able to declare a value (for any alternative) that is no higher than 1 and 6 respectively. If more points are distributed by the system, those points will be distributed to all three users; thus the relative influence that $u_1$ gained in this round will remain.

Now consider again the possibility of $u_3$ trying to manipulate the vote as described in Section 4. By declaring $\langle 7, 0, 0 \rangle$ instead of his true preferences, $\langle 7, 4, 0 \rangle$, the outcome will be changed (as can be seen in Figure 5). The meeting is scheduled to be in the morning (while $u_1$ and $u_2$ pay no tax). However $u_3$ is now taxed by 5 points, and his gained utility becomes 2 instead of 3. Dishonesty doesn't pay!

| | **Morning** | **Mid-day** | **Evening** | **Tax** |
|---|---|---|---|---|
| $u_1$ | 0 | 0 | 5 | 0 |
| $u_2$ | 0 | 4 | 0 | 0 |
| $u_3$ | 7 | 0 | 0 | 5 |
| Sum | 7 | 4 | 5 | |

Figure 5: A Futile Manipulation of the Schedule with CTm

**Theorem 1** *Each of the above scheduling systems, with the embedded CTm procedure, is non-manipulable and maintains all the other desirable attributes of the original mechanism.*

**Proof.**

To show that the mechanisms are non-manipulable we need to show that at each scheduling step each user's dominant strategy is to truly state his convenience for each possible schedule (no matter what the others do). In other words, we need to show that declaring $\vec{c_i^k}$ is the dominant strategy. Thus we have to show that user $u_i$'s benefit from declaring $\vec{c_i^k}$ is greater than any other declaration $\vec{d_i^k}$:

$b_i^k(f(D_{\neg i}^k, \vec{c_i^k})) - b_i^k(f(D_{\neg i}^k, \vec{d_i^k})) =$

Expanding the benefit $b$ into convenience minus tax, we get:
$c_i(f(D_{\neg i}^k, \vec{c_i^k})) - t_i^k(f(D_{\neg i}^k, \vec{c_i^k})) - c_i(f(D_{\neg i}^k, \vec{d_i^k})) + t_i^k(f(D_{\neg i}^k, \vec{d_i^k})) =$

Expanding the tax $t$ into its components, we get:
$c_i(f(D_{\neg i}^k, \vec{c_i^k})) - \sum_{j\neq i} d_j^k(f(D_{\neg i}^k)) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{c_i^k}))$
$-c_i(f(D_{\neg i}^k, \vec{d_i^k})) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k)) - \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{d_i^k})) =$

Eliminating equivalent plus and minus terms and adding and subtracting $c_i(s_0)$, we get:
$c_i(f(D_{\neg i}^k, \vec{c_i^k})) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{c_i^k})) - [c_i(f(D_{\neg i}^k, \vec{d_i^k})) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{d_i^k}))] =$

By $c_i^k$'s definition:
$c_i(f(D_{\neg i}^k, \vec{c_i^k})) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{c_i^k})) - [c_i(f(D_{\neg i}^k, \vec{d_i^k})) + \sum_{j\neq i} d_j^k(f(D_{\neg i}^k, \vec{d_i^k}))] =$

Let $\hat{s}^k$ be the schedule that maximizes $c_i^k(s^k) + \sum_{j\neq i} d_j^k(s)$ (i.e., $\hat{s}^k = f(D_{\neg i}^k, \vec{c_i^k})$), and let $\hat{k}^k$ be the schedule that maximizes $\sum_{i=1}^n d_i^k(s)$ (i.e., $\hat{k}^k = f(D_{\neg i}^k, \vec{d_i^k})$). Then (by the definition of the choice function $f$) we get:

$c_i(\hat{s}^k) + \sum_{j\neq i} d_j^k(\hat{s}^k) - [c_i(\hat{k}^k) + \sum_{j\neq i} d_j^k(\hat{k}^k)] =$

By $\hat{s}^k$'s definition:
$max_{k,s}[c_i(s^k) + \sum_{j\neq i} d_j^k(s^k)] - [c_i(\hat{k}^k) + \sum_{j\neq i} d_j^k(\hat{k}^k)] \geq 0$ $\square$

Showing that our procedure preserves the other desired attributes is straightforward. However it is worth noting that this is so due to some subtle considerations. As an example, consider the case where instead of bounding the users' statements per time slot, $((\max_{t=t^*}^T C_i^t) \leq w_i^{t^*})$, we would have bounded the *sum* of the points $((\sum_{t=t^*}^T C_i^t) \leq w_i^{t^*})$.

In such a case, our procedure would maintain all the desired attributes, but one, Independence of Irrelevant Alternatives. The reason is that by bounding the entire sum, we make the spread of convenience points be dependent on the number of time-slots. As an example, consider a scenario where $u_1$ strongly prefers evening over the other two possibilities, $u_3$ prefers mornings, while $u_2$ is indifferent between evening and mid-day but prefers both to morning. Assume that the wealth of the users is, respectively, 69, 20, 80. Thus the preference profile will become as shown in Figure 6. Morning will be chosen.

| | **Morning** | **Mid-day** | **Evening** | **Tax** |
|---|---|---|---|---|
| $u_1$ | 0 | 0 | 69 | 0 |
| $u_2$ | 0 | 10 | 10 | 0 |
| $u_3$ | 80 | 0 | 0 | 79 |
| Sum | 80 | 10 | 79 | |

Figure 6: Vote with Three Alternatives

However, if alternative mid-day is taken out, $u_2$ will be able to (truly) change his preference in the way described in Figure 7. A different schedule will be chosen, even though a non-winning alternative (mid-day) had been removed.

| | **Morning** | **Evening** | **Tax** |
|---|---|---|---|
| $u_1$ | 0 | 69 | 60 |
| $u_2$ | 0 | 20 | 11 |
| $u_3$ | 80 | 0 | 0 |
| Sum | 80 | 89 | |

Figure 7: Vote with Two Alternatives

As can be seen in the following section, maintaining the attribute of Independence of Irrelevant Alternatives in our mechanism is especially important if we want to give the users more privacy at the expense of the powerful concept of equilibrium in a dominant strategy.

# 6 More Privacy means Less Stability

In the calendar oriented scheduling mechanism presented above, the system has direct access to each user's calendar. This is due to the centralized meeting scheduling process. In such a mechanism, the users participate in the process by rating the available time slots in their calendars. The users have no active role in the scheduling of a given meeting.

In the meeting oriented and the schedule oriented scheduling, the system also has direct access to each user's calendar, even though the users have an active role in the schedule process itself. The main reason for that access is to maintain the stability of the system. It does not allow a user to hide an available time slot and perhaps benefit from it.

In this section, we present yet another meeting oriented scheduling mechanism that gives more privacy to the users. As a result, the equilibrium point will shift from the *dominant strategy equilibrium* to the more common *Nash equilibrium*.

Instead of having direct access to each user's calendar, the system will be able to distribute signed receipts. A receipt, $R^i_j$, has the following semantics:

`To whom it may concern:`

```
   I, the scheduling system, declare
that I have to schedule a meeting for
user u_i in time slot t_j.

   The System
```

A digital signature cryptosystem (like the one described in [24]) will be used to ensure that no one is able to generate the system's electronic signature. However, every one can read the receipt and verify that the system did sign it.

Using the digital signature cryptosystem we, can now define the following variant of the meeting oriented scheduling mechanism:

- The initiator of the meeting sends all the participants a message that includes the time window of the meeting.
- Each user declares its available slots within that time window. The *union* of time-slots declared $\mathcal{T}$ is considered for the meeting.
- Each user, $i$, that has a meeting in one of the time slots $t_j$ in $\mathcal{T}$ shows the system's receipt $R^i_j$ for that time slot, and this time slot is eliminated from the set (i.e., $\mathcal{T} = \mathcal{T} - t_j$).
- If the resulting set $\mathcal{T}$ is empty, then the meeting is rejected by the system, and the initiator should retry with a larger time slot. Otherwise, each user rates all time slots in $\mathcal{T}$ such that the maximal value is bounded by the number of points the user has. The CTm is used to choose the time slot for the meeting and to tax the users.

**Theorem 2** *The combined strategy of declaring the true set of available slots within the time window of the meeting is in Nash equilibrium.*

**Proof.** We need to show that under the assumption that all other users are using this strategy (i.e., declaring their true set of available slots), $u_i$ cannot benefit by declaring something different from his true available slots.

There are two ways in which $u_i$ can misrepresent his true available time slots for the meeting:

1. Pretend to have a meeting, by not declaring one of the available time slots.
2. Pretend to be available, by declaring a time slot in which a meeting was already scheduled for him.

We will show that these two kinds of deception are not beneficial.

1. If $u_i$ hides an available slot $s$ than there are two cases:
   (a) $s$ is available to all other users. In this case, under the assumption that all other users are telling the truth, $s$ will be declared. $u_i$ will not be able to eliminate it with a system receipt (since he does not have one). He will need to rate $s$ just as he would have had to after declaring the truth.

(b) $s$ is not available for one or more other users. Is this case, $s$ will not be included in the final set regardless of $u_i$'s declaration. He would not have to rate it anyway.

2. If $u_i$ declares a slot that is not available to him then there are also two cases:

   (a) If that slot is not included in the final set, then it does not change anything.

   (b) Otherwise, due to Nash's Independence of Irrelevant Alternatives (which is satisfied by our CTm-based mechanism), this extra alternative has no effect, unless it is the one that will be chosen. This means that it either has no influence on the outcome, or that $u_i$ will have to attend two meetings at the same time. $\square$

Note that by adopting the Nash Equilibrium point, we can more easily adjust our schedulers to deal with *Open Systems*. The only adjustment that is needed is to find an adequate way to compare the points (or declarations) that are given by different users (from independent systems). An example of a plausible method of comparison is presented in the following section.

## 7 Hierarchy of Importance

Up to now we have assumed that all users have equal influence on the scheduling system. It seems reasonable, however, that the preferences of certain users (e.g., managers) might carry greater weight in deciding when to set a meeting.

Fortunately, the influence of a user on the social decision may be easily controlled without losing the power of the mechanism as an effective preference revealer [7] (of course, by assigning weights to users we lose the possibility of anonymity). This control may be achieved by giving an influence weight $z_i (\in \mathbb{R}^+)$ to each user $u_i$. Given $u_i$'s normalized preferences $\vec{v_i}$, each $v_i(s_k)$ is divided by $z_i$; only then is the choice function invoked. Then, to keep *truth* as the dominant strategy, the calculated tax (according to the weighted votes) is multiplied by $z_i$.

**Lemma 1 (due to I. J. Good)** *Even when influence weights are used along with user preferences, it is still user i's best strategy, at each step k of the procedure, to vote over the alternatives at that step ($\mathcal{S}^k$) according to his* true *preferences.*

## 8 Dealing with Soft Constraints

One advantage of seeking the maximization of the system's *global* utility, as done in the scheduling procedures presented above, is that many other considerations of the scheduling process may be incorporated into the decision process in a unified way. For example, there may be other factors that may influence the system's global convenience for a given meeting, such as the place it will take place, the audio-visual equipment that it will use, etc.

In some scheduling systems, these considerations are treated separately, apart from the process that chooses among alternatives. Constraints such as the location of a meeting are "hard," in that they are either satisfied or not. In our schedulers, on the other hand, such constraints may be soft, so as to derive a more efficient global decision.

To establish this highly desirable phenomenon, all that the system administrator needs to do is to assign these resources/constraints convenience points as well (e.g., a big room will be more convenient for having a meeting with lots of people, a powerful overhead projector is more convenient in a partially lit room). Then the process should also maximize over the convenience "declarations" of these constraints. The only fundamental difference in the treatment of resources/constraints is that no tax may be levied on them.

## 9 Conclusions

We have introduced three scheduling mechanisms for setting up meetings in Closed Systems. All three mechanisms make use of primitive economic markets, where users assign "convenience points" to indicate their preferences over alternatives. The points for each alternative are then examined to establish the group decision that maximizes utility.

The first of the three mechanisms was Calendar Oriented Scheduling, where users assign convenience points to available time slots. The second mechanism was the Meetings Oriented approach, where users assign points to combinations of meetings and time slots (i.e., a specific meeting with specific individuals at a specific time). The third mechanism was Schedule Oriented Scheduling, where users express preferences over entire schedules (each schedule identifying one possible ordering of all future meetings). The mechanisms have varying degrees of computational complexity, and find more or less optimal schedules, but they all approximate the optimal utilitarian choice, and find a solution that is Pareto Optimal and is the Condorcet Winner.

All three mechanisms, however, are manipulable. We thus introduced the Clarke Tax as a method for removing manipulability from the above three scheduling mechanisms. It was shown that using this taxation technique, we are able to maintain all the previous positive attributes of the original mechanisms, while removing manipulability. We also showed how additional user privacy could be maintained, at the cost of decreased stability of the system, how users of varying influence could be incorporated into the scheduling system, and how soft constraints could be handled.

## Acknowledgments

This work has been supported in part by the Air Force Office of Scientific Research (Contract F49620-92-J-0422), by the Rome Laboratory (RL) of the Air Force Material Command and the Defense Advanced Research Projects Agency (Contract F30602-93-C-0038), by an NSF Young Investigator's Award (IRI-9258392) to Prof. Martha Pollack, and by the Israeli Ministry of Science and Technology (Grant 032-8284).

## References

[1] J. Doyle and M. P. Wellman. Impediments to universal preference-based default theories. *Artificial Intelligence*, 49(1–3), 1992.

[2] E. Ephrati and J. S. Rosenschein. The Clarke Tax as a consensus mechanism among automated agents. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 173–178, Anaheim, California, July 1991.

[3] E. Ephrati and J. S. Rosenschein. Multi-agent planning as search for a consensus that maximizes social welfare. In *Pre-Proceedings of the Fourth European Workshop on Modeling Autonomous Agents in a Multi-Agent World*, Rome, Italy, July 1992.

[4] E. Ephrati and J. S. Rosenschein. Reaching agreement through partial revelation of preferences. In *Proceedings of the Tenth European Conference on Artificial Intelligence*, pages 229–233, Vienna, Austria, August 1992.

[5] E. Ephrati and J. S. Rosenschein. Distributed consensus mechanisms for self-interested heterogeneous agents. In *Proceedings of the First International Conference on Intelligent and Cooperative Information Systems*, pages 71–79, Rotterdam, The Netherlands, May 1993.

[6] P. C. Fishburn. Condorcet social choice functions. *SIAM Journal on Applied Mathematics*, 33:469–489, 1977.

[7] I. J. Good. Justice in voting by demand revelation. *Public Choice*, 29(2):65–70, 1977.

[8] J. Green, E. Kohlberg, and J. J. Laffont. Partial equilibrium approach to the free rider problem. *Journal of Public Economics*, 6:375–394, 1976.

[9] Theodor Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.

[10] John C. Harsanyi. Utilitarian morality in a world of very half-hearted altruists. In Walter P. Heller, Ross M. Starr, and David A. Starrett, editors, *Social choice and public decision making*, chapter 3, pages 57–73. Cambridge University Press, Cambridge, London, 1986.

[11] L. Hurwics. On informationally decentralized systems. In R. Rander and C.B. McQuire, editors, *Decision an organization*. North-Holland, 1972.

[12] J. J. Laffont and E. Maskin. The theory of incentives: an overview. In Hildenbrand Werner, editor, *Advances in Economic theory*, chapter 3, pages 31–94. Cambridge University Press, Cambridge, 1980.

[13] R. Duncan Luce and Howard Raiffa. *Games and Decisions*. John Wiley & Sons, New York, 1957.

[14] R.P. Mohanty and M.K. Siddiq. Multiple projects-multiple resources-constrained scheduling: Some studies. *IJPR*, 27(2):261–280, 1989.

[15] Herve Moulin. Characterization of the pivotal mechanism. *Journal of Public Economics*, 31:53–78, 1986.

[16] J. F. Nash. The bargaining problem. *Econometrica*, 28:155–162, 1950.

[17] J. F. Nash. Two-person cooperative games. *Econometrica*, 21:128–140, 1953.

[18] S.J. Noronha and V.V.S. Sarma. Knowledge-based approaches for scheduling problems: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 3(2):160–171, June 1991.

[19] Bezalel Peleg. *Game Theoretic Analysis of Voting in Committees*. Cambridge University Press, Cambridge, 1984.

[20] J.H. Peterson. A comparison of exact approaches for solving the multiple constrained resource project scheduling problem. *Management Science*, 30(7):854–867, 1984.

[21] Krithi Ramamritham, John A. Stankovic, and Wei Zhao. Distributed scheduling of tasks with deadlines and resource requirements. *IEEE Transactions on Computers*, 38(8):1110–1123, August 1989.

[22] Anatol Rapoport and M. Guyer. A taxonomy of $2 \times 2$ games. *Yearbook of the Society for General Systems Research*, XI:203–214, 1966.

[23] J. Rawls. *A Theory of Justice*. Belknap, Cambridge, MA, 1971.

[24] R. L. Rivest, A. Shamir, and Adleman L. A method for obtaining digital signatures and bublic key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[25] S. Sandip and E. Durfee. A formal analysis of communication and commitment in distributed meeting scheduling. In *Proceedings of the Eleventh International Workshop on Distributed Artificial Intelligence*, pages 333–344, Glen Arbor, Michigan, February 1992.

[26] Sandip Sen and Edmund H. Durfee. On the design of an adaptive meeting scheduler. In *Proc. of the Tenth IEEE Conference on AI Applications*, March 1994. (to appear).

[27] Philip D. Straffin, Jr. *Topics in the Theory of Voting.* The UMAP Expository Monograph Series. Birkhäuser, Boston, 1980.

[28] T. N. Tideman and G. Tullock. A new and superior process for making social choice. *Journal of Political Economy*, 84(6):1145–1159, 1976.

[29] G. Tullock. The demand-revealing process as a welfare indicator. *Public Choice*, 29(2):51–63, 1977.

[30] G. Zlotkin and J. S. Rosenschein. Negotiation and task sharing among autonomous agents in cooperative domains. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 912–917, Detroit, Michigan, August 1989.

[31] G. Zlotkin and J. S. Rosenschein. A domain theory for task oriented negotiation. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 417–42, Chambery, France, August 1993.

[32] G. Zlotkin and J. S. Rosenschein. Negotiation with incomplete information about worth: Strict versus tolerant mechanisms. In *Proceedings of the International Conference on Intelligent and Cooperative Information Systems*, pages 175–184, Rotterdam, May 1993.