Evaluation of Automatically Designed Mechanisms

Anthony Jameson, Christopher Hackl and **Thomas Kleinbauer**
DFKI, German Research Center for Artificial Intelligence
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
<first name>.<last name>@dfki.de

Abstract

In their UAI 2002 paper "Complexity of Mechanism Design", Conitzer and Sandholm presented computationally tractable methods for automated mechanism design (AMD): For a given specific preference aggregation problem, such a method generates a nonmanipulable outcome selection mechanism. The present paper examines some of the decisions that need to be made by a designer who is considering the application of AMD in a particular context. Using examples from the application area of rating aggregation, we discuss several potentially desirable properties of mechanisms: symmetry, determinism, familiarity, equity, and robustness. Our example evaluations illustrate that some of these properties can be straightforwardly achieved with the methods of AMD, whereas others may sometimes require the designer to resort instead to standard mechanisms (such as averaging of ratings) that may not fall within the scope of AMD. We conclude that it is advisable, in any particular application setting, to conduct comparative evaluations of potential mechanisms; and that these comparisons should sometimes include hand-crafted mechanisms.

1 Introduction

1.1 Nonmanipulable Aggregation of Preferences

In many practical settings, an algorithm is required that solves a preference aggregation problem: choosing an outcome on the basis of the expressed preferences of two or more agents. Research on methods for solving such problems has been conducted for decades and within several fields, including economics, political science, game theory, and artificial intelligence. Stimulated especially by Arrow's seminal book *Social Choice and Individual Values* ([1]), there has been a great deal of discussion of the desirable properties of preference aggregation mechanisms, which are often surprisingly difficult to achieve simultaneously in a single mechanism.

One property that is often desired is *nonmanipulability* (sometimes referred to with other terms, such as *incentive-compatibility*): Each agent involved should find that it is in her own interest to specify her preferences accurately; it should not be possible for an agent to bring about a more desirable outcome for herself by stating her preference *strategically* (see, e.g., [10, 7, 8]). For various particular classes of preference aggregation problem, such as particular types of auction, mechanisms have been designed that have this desirable property,

In a recent paper ([2]) Conitzer and Sandholm (referred to hereafter as "CS") recently introduced a novel approach to this general problem, *automated mechanism design* (*AMD*): As an alternative to the hand-crafting of general mechanisms, their method makes it possible to have a mechanism designed automatically which is optimal for a particular setting. CS present a number of results concerning the computational complexity of AMD. In a more recent manuscript ([3]), they discuss several examples of applications of AMD, some of which make use of advances in the methods relative to the presentation in [2].

The present paper discusses example applications in one particular type of setting. While these applications may be of interest in themselves, the main contribution of this paper is on a more general level: We show that, when considering a particular application of AMD, the designer has a number of choices to make:

1. The AMD framework itself allows the designer to make a variety of choices about the properties of the to-be-generated mechanism, even within a specific application situation. These choices can have consequences of various sorts, ranging from the expected utility that will result from the application of the mechanism to the subjective acceptability of the mechanism to any humans that are involved in the process. Some type of comparative evaluation of the possible mechanisms is therefore required. The eval-

uation may consist only in an intuitive assessment of the suitability of a given mechanism for a given application, but it may also involve quantitative analysis of the performance of the mechanism under various conditions.

2. It can also be worthwhile to consider the alternative of using simple hand-crafted mechanisms, such as the averaging of preferences that are expressed on a numerical scale.[1] Even though such a mechanism may be less desirable than an automatically generated mechanism in one or more respects, it may be found to be the most suitable one in a particular application setting.

The "evaluation of automatically designed mechanisms" referred to in the title of this paper is, therefore, not an evaluation of the entire approach of automated mechanism design—an enterprise that would exceed the scope of any single paper. The focus is, rather, on criteria for evaluating particular automatically designed mechanisms in particular application settings.

1.2 Examples of Application Settings

For concreteness, in this paper we restrict our attention to a particular type of preference aggregation problem: the aggregation of preferences that can be expressed on one-dimensional rating scales for the purpose of arriving at a joint rating by several raters. Although this class includes only a tiny fraction of the settings to which AMD can be applied, it does have considerable generality and practical significance, as the examples given below will show. And although the specific quantitative results that we report cannot be generalized to different settings, the general criteria and considerations we present are also relevant to very different application areas of AMD (e.g., the ones discussed in [3]).

In this section, we give motivating examples from three typical applications in which nonmanipulable rating aggregation is a serious issue.

Reviewers' Ratings of Conference Papers Some conference management systems now allow reviewers to (a) view, for a paper that they have reviewed, the reviews and ratings that the other reviewers have provided; and (b) change their own reviews and ratings. Although these possibilities have some advantages, they also make it especially easy for a single reviewer to manipulate the outcome of the process. Consider the example illustrated in Figure 1: Reviewer 3 originally gave a particular paper an overall rating of 3 on a scale from 1 to 5. She then sees that

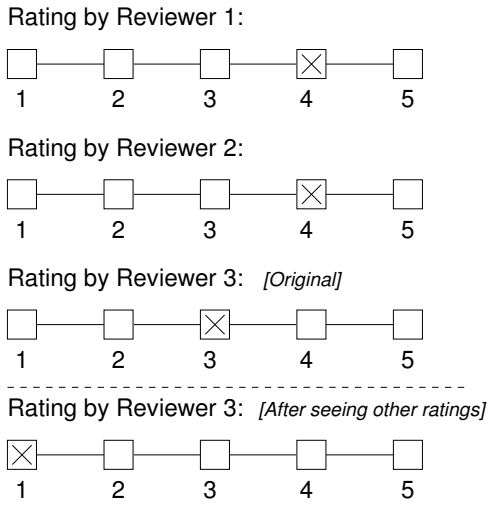

Figure 1. *Example of rating manipulation in the context of the reviewing of conference papers.*
(Discussion in text.)

Reviewers 1 and 2 both gave ratings of 4. Realizing that the average rating, after rounding up, will be 4, she may shift her own rating down to 1, thereby ensuring that the average will be the "correct" rating 3. It is an appealing idea to have a method for aggregating reviewers' ratings which, if correctly understood by the reviewers, would encourage them to specify their sincere rating of each paper's value.[2] Incidentally, this example illustrates that manipulation does not presuppose dishonesty or an exclusive focus on one's own interests: Reviewer 3 may see herself simply as ensuring that the overall rating is the "correct" one.

Preferences for Shared Music McCarthy and Anagnost ([6]) introduced the system MUSICFX, which selects a genre of music to be played in a fitness center on the basis of the stored music preferences of the members who are currently working out. For each of 91 genres (e.g., "classic rock"), each fitness center member specifies a rating on a five-point scale ranging from "I hate this music" to "I love this music". Each time a member enters or leaves the center or changes their ratings, MUSICFX applies a simple formula (taking the sum of the squared ratings) to compute an overall preference index for each genre for the set of members currently working out. The 91 indices are then mapped onto 91 probabilities that the system uses to de-

[1] The term *hand-crafted* refers in this paper to mechanisms that are chosen and/or invented by a human mechanism designer rather than automatically generated by the methods of CS. A hand-crafted mechanism may therefore be a well-known standard algorithm; cf. 4.2.

[2] Viewing this particular problem as a preference aggregation problem presupposes that the goal is to arrive at an overall rating that in some sense appropriately maximizes the satisfaction of the individual reviewers. The problem can also be viewed as a belief aggregation problem, where each rating constitutes evidence concerning the "true" rating of the paper. The formal methods discussed in this paper are actually applicable to the latter conception as well, for example if the notion of *utility* is replaced by the logarithm of the likelihood of an observed rating given a particular "true" rating.

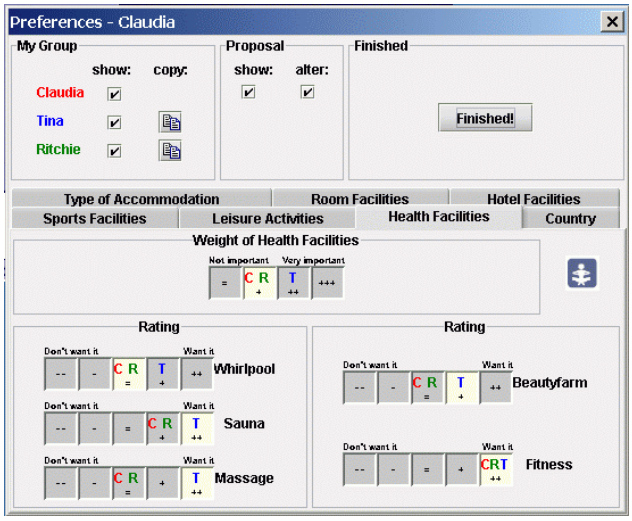

Figure 2. *Dialog box for the collaborative specification of preferences.*
(The currently active group member is Claudia. The preferences of each member are represented by a uniquely colored letter; Claudia cannot change the position of the letters "T" and "R"..)

termine which genre to play at any given moment. During the initial trial period, McCarthy and Anagnost actually observed members trying (with varying degrees of success) to manipulate the system by changing their preference specifications as a function of the music currently being played and the other members who were present.

Multi-Attribute Preferences in a Group Recommender System The research group of the present authors has developed a system (described in [4]) that recommends products and services to a group of users whose members can communicate only asynchronously via the web-based interface. Like the web-based system ACTIVE BUYER'S GUIDE (http://www.activebuyerguide.com/) and the once-popular system PERSONALOGIC (which is no longer available), this system requires each user to specify preferences concerning several attributes for each of a number of value dimensions. A novel aspect of the dialog box for specifying preferences (see Figure 2) is the way in which it allows the current user to see any previously specified preferences of other group members—and to copy them conveniently if she so desires. This feature has the advantage of allowing users to minimize the amount of tedious rating work by copying (and postediting) ratings of a group member with similar preferences. But it also opens the door to manipulation in the same way as with the reviewing problem.

2 Automated Mechanism Design

We reproduce here only the key notation and concepts of CS that are necessary for the understanding of the present paper.

A *preference aggregation setting* includes a set of outcomes O and a set of agents A with $|A| = N$. In the context of rating aggregation, the outcomes are the possible aggregate ratings.

In addition, the preference aggregation setting includes, for each agent, the following elements:

1. A set of *types* Θ^i
   In the context of rating aggregation, this corresponds to the set of possible ratings (e.g., the five values on a scale). The *true type* of an agent is the rating that corresponds to that agent's true preference.
2. A probability distribution p_i over Θ^i
   This distribution represents an a priori expectation about the true type of the agent.
3. A utility function u_i: $\Theta^i \times O \rightarrow \mathbb{R}$
   This function indicates how (dis)satisfied the agent will be if a particular outcome is selected.

A *randomized mechanism* for a preference aggregation setting is a function that, given any vector of *reported types*, produces a probability distribution over the outcome set. That is, it is a function $p : \Theta^1 \times \Theta^2 \times \ldots \Theta^N \rightarrow \mathcal{PROBDISTS}(O)$. [3] Considering for concreteness the case where there are only two agents, the mechanism can be seen as a three-dimensional matrix of probabilities p_{ij}^k, where p_{ij}^k is the probability of choosing the outcome o_k when the reported types are θ_i^1 and θ_j^2.

With regard to the criterion of nonmanipulability, CS consider two solution concepts, *dominant strategies* and *Bayes-Nash equilibrium*. We restrict our attention to the former concept, which is much more robust and therefore better applicable in settings such as the ones introduced in 1.2. In particular, the dominant-strategies concept is applicable even when one rater does not know the prior probabilities for the true types of the other rater(s) but does know what preference(s) one or more other raters have specified. Specifically, a mechanism is said to *implement its outcome function in dominant strategies* if truth telling is always optimal even when the types reported by the other agents are already known. CS show how this condition can be expressed in terms of constraints on the values of the p_{ij}^k that constitute the mechanism. One such constraint states that, whatever type Agent 2 may report, the expected utility of the outcome will be highest for Agent 1 if he reports his true type θ_i^1:

[3] CS also consider purely deterministic mechanisms, but they show that the more general class of randomized mechanisms has advantages in terms of both (a) the computational complexity of finding a suitable mechanism and (b) the utility yielded by the mechanisms. Accordingly, we will consider the broader class in this paper. But as we will see in Section 3, determinism can be a desirable property of a mechanism in some application settings. In [3], Conitzer and Sandholm include examples of automatically designed mechanisms that were explicitly constrained to be deterministic.

For every $\theta_j^2 \in \Theta^2$, for every $\theta_i^1, \theta_l^1 \in \Theta^1$:

$$\sum_{k:o_k \in O} p_{ij}^k u_1(\theta_i^1, o_k) \geq \sum_{k:o_k \in O} p_{lj}^k u_1(\theta_i^1, o_k). \quad (1)$$

There is of course an analogous constraint concerning the type reported by Agent 2.

It is trivial to find just any mechanism that satisfies these constraints. For example, a mechanism that always yields the same outcome regardless of the reported types, or that always yields the outcome corresponding to the type reported by Agent 1, is nonmanipulable. Typically we will want to find a mechanism that maximizes the expected value of some *objective function* $g(\theta_i^1, \theta_j^2, o_k)$ (e.g., the sum of the utilities of the outcome for the individual agents). So mechanism design can be seen as the optimization problem of finding a matrix of p_{ij}^k that fulfill the nonmanipulability constraints while maximizing the expression:

$$\sum_{i:\theta_i^1 \in \Theta^1} \sum_{j:\theta_j^2 \in \Theta^2} \sum_{k:o_k \in O} p_1(\theta_i^1) p_2(\theta_j^2) p_{ij}^k g(\theta_i^1, \theta_j^2, o_k). \quad (2)$$

As long as the objective function g is polynomially computable, this problem can be solved in polynomial time by linear programming.

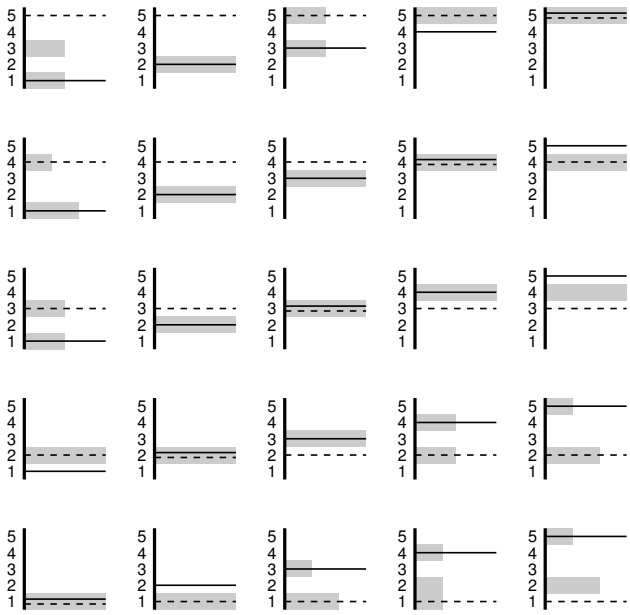

Figure 3. *Example of an automatically designed nonmanipulable mechanism.*
(Each histogram shows the probability distribution over outcomes for a particular combination of ratings by two raters; the rating of Rater 1 is indicated with a dashed line, that of Rater 2 with a solid line.)

3 Desirable Properties of Mechanisms

Although AMD ensures that an automatically designed mechanism will maximize the expected value of the objective function, given the other constraints specified, there are other potentially desirable properties of mechanisms. All of these properties have been discussed in relevant previous literature (e.g., on social choice and mechanism design). In this section, we will introduce and discuss these properties concretely with reference to the example problem of aggregating reviewers' ratings. Our goal is not to add new general insights concerning these properties but to show how they can be taken into account by a designer who is considering using AMD. We will see that some of the properties can be dealt with straightforwardly within AMD, while others are more problematic.

As a concrete illustration, we first display a mechanism designed automatically with the method of CS without regard to any of these desirable properties. Figure 3 visualizes a mechanism that was generated with 2 agents and 5 possible ratings using straightforward utility functions and objective functions.[4] It can be verified visually that the mechanism is nonmanipulable for both agents, even if the rating of the other agent is known. But the mechanism seems unlikely to be adopted, for example, by any conference program chair. Its drawbacks can be formulated in terms of some desirable properties of mechanisms:

Symmetry This particular mechanism exhibits several instances of asymmetry.[5] For example, when Rater 1 specifies "4" and Rater 2 "5", the outcome is "4"; but when Rater 1 specifies "5" and Rater 2 "4" the outcome is "5". It would presumably be hard to persuade Rater 2 that this type of pattern was justifiable, in the absence of any relevant difference between the two raters (e.g., a greater importance of Rater 1, which would be reflected in an asymmetric objective function). Even if it is not obvious what particular disadvantage anyone suffers from an asymmetric mechanism, there are presumably at least some situations in which a symmetric mechanism will be more readily accepted.[6]

There is also a technical advantage of symmetry: The symmetry constraint implies that many pairs of probabilities within the matrix for a mechanism must be equal, and this fact can be exploited in the formulation of the linear optimization problem so as to reduce the number of variables

[4] In the terms introduced in Section 4.1 below, this mechanism was generated with the linear utility function, without regard to equity, with nonuniform prior expectations, and without the symmetry constraint. This mechanism is compared with several others in Table 2.

[5] A formalization of the concept of *symmetry* in this context will be given in 4.1.

[6] In social choice theory, the same basic criterion is captured by the notion of *anonymity*, or invariance under permutation of the agents involved.

significantly. As a result, settings involving larger numbers of agents and/or outcomes can be dealt with if the symmetry constraint is introduced.

Determinism Nondeterministic mechanisms like coin-flipping are widely accepted in some situations. But a probability distribution like the one at the bottom right in Figure 3 has an arbitrary appearance that could diminish its acceptability to humans.

Perhaps more importantly, the need to perform some sort of random drawing to determine the outcome imposes what may be serious practical constraints: This event (e.g., a roll of dice by a program chair) will have to occur at some particular point in time. Before that point, no-one will know what the outcome is. And after that point, changes to preference specifications should no longer be allowed, even if it is feasible to repeat the random drawing process. For if subsequent changes are allowed, a rater may be able to manipulate the outcome by changing her rating slightly and hoping for better luck in the next draw. Even though her expected utility on the second draw cannot be higher than her expected utility on the first draw was (by the definition of nonmanipulability), it may well be higher than the actual utility that she derived from the first draw.

Therefore, in some settings it may be desirable to restrict the automatically generated mechanisms to those which are deterministic, even if this restriction entails an increase in computational cost and/or a decrease in the expected value of the objective function. Some examples of mechanisms generated in this way are given by Conitzer and Sandholm in [3].

Familiarity Even if Figure 3 showed a symmetric pattern of entries involving only probabilities of 1.0 and 0.0, the mechanism would not necessarily be easy for people to understand, remember, or apply. By contrast, familiar aggregation methods like averaging (perhaps with rounding) or taking the median rating can often be applied by anyone who knows the ratings; there is then no need to consult an external source in order to find out what the outcome is. Even if the familiar method is manipulable (as is the case with averaging), its familiarity may in some situations constitute enough of an advantage to justify its use.

The criterion of familiarity is the most elusive of the criteria discussed here. A more detailed analysis could break it down into several more specific and formalizable criteria. For example, one could attempt to predict on the basis of general principles how easily a human agent could perform each of the following tasks using particular resources (e.g., performing only mental arithmetic): computing the outcome for particular preference specifications; verifying that the mechanism is nonmanipulable; and verifying that the mechanism tends to yield outcomes that are desirable according to the objective function. Ideally, we would want to be able not only to evaluate the familiarity of a given mechanism but also to constrain the mechanism design process so that it tends to generate familiar mechanisms—as can be done for the criteria of symmetry and determinism, which can be seen as specific factors that contribute to familiarity.

Equity In many of the cells of Figure 3, we see that the outcome is more favorable for one of the raters than for the other one. For example, when Agent 1 specifies "4" and Agent 2 "2", the outcome is "2". The intermediate outcome "3" would yield exactly the same total utility according to the utility function that this mechanism presupposes, while being more equitable—and thus presumably more acceptable to human participants in some situations.

There have been many discussions of ways of defining equity and of the role that it should play in social choice problems (see, e.g., [5, 9]). In the present context, the main issue is how well equity can be taken into account within AMD. As we will see in 4.1, with some definitions of the concept, it is straightforward to take equity into account by including an index of equity as a component of the objective function, whereas other definitions cannot be so straightforwardly realized.

A more specific question in any given context concerns the extent to which an attempt to increase equity will require sacrifices with respect to either total utility or the other desirable properties of mechanisms just discussed. Answering this question may require quantitative simulations like the ones presented in Section 5.

Robustness Although the mechanism in Figure 3 is guaranteed to be nonmanipulable by agents whose utility function is exactly the one that was assumed when the mechanism was generated, it might be manipulable if one or both of the agents has a slightly different utility function (e.g., one which is concave or convex instead of linear; cf. Section 4.1). By contrast, some hand-crafted mechanisms (e.g., the median mechanism, discussed below) make hardly any assumptions about the nature of the utility functions: They presuppose only that the single rating specified by an agent is at least as desirable to that agent as any other rating. Since in practice the form of the relevant utility functions may not be known precisely, dependence on a specific class of utility functions can be a drawback.

In principle, we could allow each agent to specify not only a single preferred rating but also the form of her entire utility function (e.g., whether it is linear, convex, or concave; cf. 4.1). But there are limits to this strategy, since in general the number of possible utility functions that an agent might have can be very large, requiring the definition of a large number of possible types for each agent. As is pointed out in [3], increasing the number of types tends to lead to rapid increases in the computational cost of AMD. Hence it will

often be necessary to make some restrictive assumptions about the form(s) of the utility functions of the agents—assumptions whose truth may be hard to guarantee.

4 Method

In this and the following section, we will look at the properties of the mechanisms generated by the method of CS in a number of different settings, comparing them with selected hand-crafted mechanisms.

4.1 Settings

Restrictive Assumptions We restrict our attention to settings, such as the ones mentioned in the examples so far, which are symmetric in terms of the assumptions made about the agents involved: The number of possible types and their a priori probability distribution are the same for all agents, as are the agents' utility functions. This assumption is reasonable when the agents are drawn unsystematically from some population.

A more arbitrary assumption is that the number of possible types is equal to the number of possible outcomes, with the following one-to-one relationship: An agent has the type θ_i iff he prefers the outcome o_i to all other outcomes. It is of course possible for the number of outcomes to be greater than the number of types (as when fractional aggregated ratings such as "4.5" are allowed), and vice-versa.

Numbers of Agents We consider only settings with 2 or 3 agents. As CS note, since the complexity of the linear optimization problem is exponential in the number of agents, their method is tractable only for small numbers of agents.

Numbers of Outcomes; Presence vs. Absence of the Symmetry Constraint The numbers of outcomes (and types) considered were the largest ones that we could handle in a reasonably short time (less than 1 minute) on a normal PC:[7]

In settings where it was feasible, we generated mechanisms both with the constraint of symmetry (cf. Section 3) and without this constraint, so as to be able to compare the results. With 2 agents, this was possible with 5 outcomes; with 3 agents, it was possible with 4 outcomes.

It was possible to handle somewhat larger numbers of outcomes if the symmetry constraint was always imposed: for 2 agents, 7 outcomes, and for 3 agents, 5 outcomes.

The symmetry constraint can be defined as follows for 2 agents, with the obvious generalization for 3 agents:

$$\forall_{i,j} p_{ij}^k = p_{ji}^k. \tag{3}$$

Utility Functions We used three representative utility functions for the agents, defined as follows:

- Linear: $u(\theta_i, o_k) = -|(o_i - o_k)|$
- Concave: $u(\theta_i, o_k) = -(o_i - o_k)^2$
- Convex: $u(\theta_i, o_k) = -\ln(1 + |(o_i - o_k)|)$

Thus each utility function presupposes that the dissatisfaction of an agent increases monotonically with the distance between the outcome and his own preferred outcome; the "distance" being simply the difference between the two indices.

When interpreting the results presented below, it will be helpful to bear in mind the following points concerning the outcome that maximizes the total utility of two agents who have different preferences:

- With the linear utility function, any outcome between the favored outcomes of the two agents (including either of these outcomes) will yield the same total utility.
- With the concave utility function, the utility-maximizing outcome is one that is (about) midway between the preferred outcomes of the two agents. (There may be no outcome that is exactly midway between them.)
- With the convex utility function, the utility-maximizing outcome is one that is exactly the preferred outcome of one of the agents.

Objective Functions One objective function is simply the total utility for the agents involved, defined as follows for 2 and 3 agents, respectively:

$$g_u(\theta_i^1, \theta_j^2, o_k) = u(\theta_i^1, o_k) + u(\theta_j^2, o_k). \tag{4}$$

$$g_u(\theta_i^1, \theta_j^2, \theta_l^3, o_k) = u(\theta_i^1, o_k) + u(\theta_j^2, o_k) + u(\theta_k^3, o_k). \tag{5}$$

A second component is *equity*, defined as follows for 2 and 3 agents, respectively:

$$g_e(\theta_i^1, \theta_j^2, o_k) = -|u(\theta_i^1, o_k) - u(\theta_j^2, o_k)|. \tag{6}$$

$$\begin{aligned} g_e(\theta_i^1, \theta_j^2, \theta_l^3, o_k) = &-1/3 \cdot (|(u(\theta_i^1, o_k) - u(\theta_j^2, o_k)| \\ &+ |(u(\theta_i^1, o_k) - u(\theta_l^3, o_k)| \\ &+ |(u(\theta_j^2, o_k) - u(\theta_l^3, o_k)|). \end{aligned} \tag{7}$$

Note that this index of equity captures the expected extent to which a specific outcome generated by the mechanism will be seen as equitable. According to a more global conception of equity, a mechanism is equitable to the extent to

[7] For more realistic (and encouraging) indications of the size of problems that can be handled by AMD, see the scalability results reported by Conitzer and Sandholm ([3]), whose implementation is more efficient than the one that we used. In our implementation, the specification of the linear programming problem described in Section 2 is built up by a JAVA program and solved by the Cassowary solver (http://www.cs.washington.edu/research/constraints/cassowary/).

which, a priori, the expected value of an outcome is equal for all agents. This conception of equity cannot be encoded in the objective function, because it involves a weighted averaging over all possible preference specifications and outcomes. (This type of equity is, however, guaranteed if the mechanism is symmetric.)

Since there is little point in maximizing equity without regard to utility, the second objective function used for the generation of mechanisms, referred to below as *utility + equity*, is defined as $g_u + g_e$. It may of course make sense to assign different weights to utility and equity, but the results reported below will give some idea of what would happen with different weights as well.

Prior Expectations Concerning Types In many practical rating settings, (e.g., the one shown in Figure 2 and, to a lesser extent, the one shown in Figure 1), preferences in the upper half of the scale are much more common than those in the lower half. With this type of preference distribution, a relatively high proportion of low ratings is likely to be due to manipulation, if manipulation is possible. We therefore chose the following (rather extreme) nonuniform distributions for our experiments:

- With 4 outcomes: [.05 .05 .45 .45]
- With 5 outcomes: [.05 .05 .30 .30 .30]
- With 7 outcomes: [.03 .03 .04 .225 .225 .225 .225]

We also generated and analyzed corresponding mechanisms based on uniform distributions. Since the results are largely similar to those for the nonuniform priors, we will refer to the results for uniform priors only occasionally.

4.2 Hand-Crafted Mechanisms

For each setting, we compared the automatically generated mechanisms with the following hand-crafted mechanisms:

Average To enhance comparability with the automatically generated mechanisms, we define the averaging mechanism in such a way that noninteger outcomes are avoided: A noninteger average is rounded to the nearest integer, unless its fractional part is 0.5, in which case one of the two neighboring integers is chosen with equal probability.

This mechanism has the appealing properties of symmetry, familiarity, and applicability to any number of agents. Moreover, it tends to maximize total utility (aside from the rounding off) if the concave utility function defined above is used. On the down side, it is manipulable (as was illustrated by the first example in 1.2). Therefore, in addition to computing results on the assumption that the agents specify their preferences accurately, we will require a mechanism that simulates manipulation:

Manipulated Average We will not try to examine all possible forms of manipulation, including, for example, cases where two agents form a coalition against the third agent. One simple case is the one illustrated in Figure 1: (a) the first $N-1$ agents specify their preferences sincerely, and (b) the Nth agent sees their ratings and gives the rating that maximizes her own expected utility.[8] We defined for each setting a mechanism *manipulated average*: Given the true types of the agents, it generates exactly the same outcomes that would be generated if the Nth agent manipulated in the way just described. We would expect this mechanism to yield worse results than the other mechanisms according to some criteria; but even so it is of interest to determine how much worse the results really are in a particular setting.

Median For two agents, this mechanism simply chooses one of the two specified ratings randomly with a probability of 0.5 and takes that rating as the outcome. It is easy to see that this mechanism is nonmanipulable for any utility function that fulfills the assumption specified at the beginning of 4.1. For concreteness, the mechanism will be called the "coin flip" mechanism when only two agents are involved.

For three agents, the median takes as the outcome a rating "in the middle" (i.e., such that one other rating is equal or higher and the remaining rating is equal or lower). It can be checked that this mechanism is likewise nonmanipulable. It is also deterministic, symmetric, and reasonably familiar. Its more general definition is applicable to any number of agents (though with an even number of agents a random choice between the two "middle" outcomes is required if they are not equal).

5 Results

Instead of presenting the results of the computational experiments as empirical results, we will discuss them in terms of general concepts and principles, referring to the data as concrete illustrations. Quantitative results that cannot be understood in this way should be viewed as illustrations of phenomena that can arise with some particular parameter settings (e.g., the fact that the manipulated-averaging mechanism performs competitively in some of the settings presented below). Accordingly, we do not present the results of statistical tests.

[8] Note that cases where more than one agent manipulates do not necessarily lead to outcomes that are farther away from the outcome yielded by truthful specification: For example, two agents might distort their ratings in opposite directions, giving rise to the same average rating as in the case of truthful specifications.

Table 1. *Properties of mechanisms in settings with 2 agents and 7 outcomes.*
(Each number shows the expected value of (a component of) the objective function for a particular mechanism. See also the explanation in the text.)

	Generated		Hand–Crafted		
Criterion	For Utility	For Utility + Equity	Average	Average (Manipulated)	Coin Flip
Linear Utility:					
Utility	−0.82	−0.83	−0.82	−0.82	−0.82
Equity	−1.49	−0.88	−0.50	−0.99	−1.63
Sum	−2.30	−1.71	−1.32	−1.80	−2.45
Concave Utility:					
Utility	−1.50	−1.51	−1.24	−1.55	−2.23
Equity	−2.09	−1.78	−0.90	−2.09	−4.47
Sum	−3.59	−3.29	−2.14	−3.65	−6.70
Convex Utility:					
Utility	−0.42	−0.48	−0.51	−0.47	−0.42
Equity	−0.84	−0.49	−0.29	−0.54	−0.84
Sum	−1.25	−0.97	−0.80	−1.02	−1.25
		det			

5.1 Two Agents, Seven Outcomes

Table 1 shows the results for the mechanisms for 2 agents and 7 outcomes, which were constrained to be symmetric.[9]

The first three lines of results can be read as follows: Using the linear utility function, we generated two mechanisms, the first one maximizing utility and the second one maximizing the sum of utility and equity. For each mechanism, the table shows the expected value of utility, equity, and their sum, so that any tradeoffs between the criteria can be seen. The three right-hand columns show the corresponding results for the three hand-crafted mechanisms. The symbol *det* below the results for a generated mechanism indicates that it is deterministic; this is the case for only one of the mechanisms in this table.

5.1.1 Linear Utility Function

Looking first at the results for the linear utility function, we see that all mechanisms yield the same total expected utility—understandably, given the nature of the utility function (cf. 4.1).[10] The mechanism that was optimized for utility + equity manages to achieve much greater equity without sacrificing utility. But note that the canonical averaging mechanism (under the assumption of no manipulation) does even better in terms of equity; that is, because of the need to ensure nonmanipulability, the automatically designed mechanism cannot achieve as precise a balance between the preferences of the two agents as averaging can.

If Agent 2 manipulates the averaging mechanism, the average equity is slightly worse than for the best automatically designed mechanism. The amount of equity involved in this difference is the amount that arises when the outcome is 0.11 units closer to the preference of one agent than to that of the other agent. (In the corresponding comparison for uniform prior expectations about the agents' types, a similar pattern arises, but the difference is only 0.04 units of equity.) So the question arises here whether this "worst case" is sufficiently bad to justify the use of an unfamiliar nondeterministic outcome selection mechanism.

The coin-flip mechanism does poorly in terms of average equity, because it always satisfies one agent completely at the expense of the other one (unless they happen to agree).

5.1.2 Concave Utility Function

Turning to the results for the concave utility function, we see that with the two automatically generated mechanisms a considerable gain in equity can be achieved without loss in utility. The relative superiority of non-manipulated averaging is greater here than it was with the linear utility function, partly because in this case averaging can improve utility as well as equity.

5.1.3 Convex Utility Function

With the convex utility function, the canonical coin-flip mechanism achieves the maximum possible utility—as does any other mechanism that ensures that at least one agent's preference is fully satisfied. One of these other mechanisms is generated automatically when utility is maximized: Although the performance of the leftmost mechanism at the bottom of the table is identical to that of the coin-flip mechanism, inspection of its probabilities reveals many seemingly arbitrary entries like 0.9594 and 0.2202. The same statement is true with uniform priors, and also with 5 possible outcomes (Table 2). It would not be straightforward to reformulate the optimization problem in such a way that only the more comprehensible probabilities of 0.0, 0.5, and 1.0 emerged. So this is a case where desirable properties like familiarity and robustness can be achieved only if the automatic mechanism generator is supplemented with human insight.

As the results for the two generated mechanisms illustrate, with this type of utility function there is inevitably a tradeoff between total utility and equity; so we can no longer gain equity at no cost simply by specifying the optimization problem carefully; instead, we have to think about the relative importance of utility and equity in the specific application scenario.

[9] When interpreting these results, one should bear in mind that we are assuming a situation in which the second agent may be aware of the rating given by the first agent; cf. Section 2.

[10] Differences of just .01 should not be interpreted, since they can arise through rounding errors.

Table 2. *Properties of mechanisms in settings with 2 agents and 5 outcomes.* (Explanation in text.)

	Generated				Hand–Crafted		
	For Utility		For Utility+ Equity				
Criterion	Sym –	Sym +	Sym –	Sym +	Average	Average (Manipulated)	Coin Flip
Linear Utility:							
Utility	–0.59	–0.59	–0.60	–0.60	–0.59	–0.59	–0.59
Equity	–0.93	–1.12	–0.60	–0.60	–0.45	–0.70	–1.18
Sum	–1.52	–1.70	–1.21	–1.21	–1.04	–1.28	–1.76
Concave Utility:							
Utility	–0.83	–0.83	–0.83	–0.83	–0.71	–0.83	–1.19
Equity	–1.12	–0.89	–0.87	–0.87	–0.57	–1.06	–2.38
Sum	–1.94	–1.72	–1.70	–1.70	–1.28	–1.88	–3.56
	det sym						
Convex Utility:							
Utility	–0.33	–0.33	–0.38	–0.38	–0.39	–0.37	–0.33
Equity	–0.67	–0.67	–0.37	–0.37	–0.30	–0.43	–0.67
Sum	–1.00	–1.00	–0.76	–0.76	–0.69	–0.80	–1.00

5.2 Two Agents, Five Outcomes

In the settings with 2 agents and only 5 outcomes (Table 2), it was not computationally necessary to restrict the automatically designed mechanisms to being symmetric. We can therefore see here whether the mechanisms generated without the symmetry constraint (in the columns labeled "Sym –") are in any respect better than those generated with the symmetry constraint ("Sym +"). Although only 1 of the 6 mechanisms generated automatically without the symmetry constraint turned out to be symmetric (the one labeled "det sym"), the table reveals no benefits that we can attain by dropping the symmetry constraint in these settings.

5.3 Three Agents, Five Outcomes

Looking at the case of 3 agents and 5 outcomes (Table 3), we first note that there is now a tradeoff between utility and equity for all three types of utility function. The three mechanisms generated to optimize the sum of utility and equity are slightly outperformed even by the manipulated-averaging mechanism with respect to this criterion. (The same pattern arises when uniform priors are assumed.) This result reflects in part the fact that the impact of a single manipulating agent on the outcome is more limited when there are 3 agents instead of 2.

The three mechanisms generated so as to optimize only utility are fully deterministic. (The same is true when uniform priors are used.) Because the optimization process is itself partly nondeterministic, it is not obvious that *any* mechanism generated in one of these three settings will be deterministic. But the results do show that there exist deterministic mechanisms that yield the maximum expected utility—a fact that is of interest given the advantages of deterministic mechanisms discussed in Section 3.

Table 3. *Properties of mechanisms in settings with 3 agents and 5 outcomes.*

	Generated		Hand–Crafted		
Criterion	For Utility	For Utility + Equity	Average	Average (Manipulated)	Median
Linear Utility:					
Utility	–0.59	–0.77	–0.68	–0.70	–0.59
Equity	–0.93	–0.65	–0.57	–0.71	–0.93
Sum	–1.52	–1.42	–1.25	–1.41	–1.52
	det				
Concave Utility:					
Utility	–0.96	–1.02	–0.87	–0.98	–1.00
Equity	–1.35	–1.20	–0.87	–1.21	–1.72
Sum	–2.30	–2.22	–1.73	–2.19	–2.72
	det				
Convex Utility:					
Utility	–0.35	–0.48	–0.44	–0.45	–0.35
Equity	–0.55	–0.39	–0.35	–0.42	–0.55
Sum	–0.90	–0.87	–0.80	–0.87	–0.90
	det				

With the convex utility function, the mechanism generated to optimize overall utility is exactly the median mechanism. So in this case, the linear solver comes up with a mechanism which is not only optimal but also deterministic, familiar, and robust. (The same statement applies with 4 outcomes and with uniform priors.)

Table 4. *Properties of mechanisms in settings with 3 agents and 4 outcomes.*

	Generated		Hand–Crafted		
Criterion	For Utility	For Utility + Equity	Average	Average (Manipulated)	Median
Linear Utility:					
Utility	−0.38	−0.41	−0.42	−0.45	−0.38
Equity	−0.68	−0.65	−0.54	−0.57	−0.68
Sum	−1.06	−1.06	−0.96	−1.02	−1.06
	det				
Concave Utility:					
Utility	−0.54	−0.59	−0.49	−0.53	−0.54
Equity	−0.97	−0.77	−0.68	−0.74	−1.00
Sum	−1.51	−1.36	−1.17	−1.27	−1.54
	det	*det*			
Convex Utility:					
Utility	−0.25	−0.25	−0.28	−0.30	−0.25
Equity	−0.43	−0.43	−0.36	−0.37	−0.43
Sum	−0.68	−0.67	−0.64	−0.67	−0.68
	det	*det*			

5.4 Three Agents, Four Outcomes

For the settings with 3 agents and 4 outcomes, it was again possible to drop the symmetry constraint for half of the generated mechanisms. Nonetheless, each of these potentially asymmetric mechanisms turned out to be not only symmetric but also strictly identical to the corresponding mechanism generated under the symmetry constraint. (The same is true when uniform priors are assumed.) Therefore, Table 4 does not distinguish between mechanisms generated with and without the symmetry constraint.

We see that the tendency of the mechanisms generated for 3 agents to be deterministic is even stronger than it was with 5 outcomes. (The one nondeterministic mechanism contains a number of probabilities of 0.5; when uniform priors are assumed, all of the corresponding mechanisms are deterministic.)

On the whole, the mechanisms generated for 3 agents tend to rate better in terms of the acceptability criteria of symmetry, determinism, familiarity, and robustness. One way of viewing this tendency is in terms of overfitting in the case of 2 agents: With only 2 agents, there tend to be many different mechanisms that satisfy the constraint of nonmanipulability. The linear solver therefore often finds a solution with apparently arbitrary properties that happens to be optimal for the specific setting in question.

6 Conclusions

Even these limited examples of the application of AMD have illustrated some of the flexibility and power of the AMD approach. They also highlight some of the factors that need to be taken into account when AMD is applied in a particular setting. Although the generation of specific mechanisms is indeed automated, the method does not eliminate the need for careful judgment and analysis, especially in the case of mechanisms that (a) will be used repeatedly and (b) need to be accepted by the humans who are somehow involved in the aggregation process.

We have seen that it is often possible to achieve one or more of the desirable properties of symmetry, determinism, familiarity, equity, and robustness without sacrificing other desirable properties; in other cases, tradeoffs arise which call for judgments about priorities in the particular application scenario. Making well-founded decisions about these tradeoffs may require performing quantitative comparisons such as the ones described here.

In some cases, the best choice may be a mechanism that falls outside of the range of automatically generated nonmanipulable mechanisms. For example, a conference program chair may decide, after looking at analyses such as those presented above, that it is reasonable to continue using an averaging mechanism as a way of summarizing reviewers' ratings; but he or she might decide to eliminate the lowest rating category if it seems to be used relatively frequently for (perhaps unconscious) manipulation. (This move would be similar to the transition from 5 outcomes to 4 outcomes shown in Tables 3 and 4, respectively.) Even in cases of this sort, automatically generated mechanisms can serve as useful standards of comparison, in that they show what could in principle be achieved with strictly nonmanipulable mechanisms.

Acknowledgments

This research was supported by the German Ministry of Education and Research (BMB+F) under grant 01 IW 001 (project MIAU). Valuable comments on an earlier draft were made by Vincent Conitzer, but the responsibility for any flaws in this final draft remains with the authors.

References

[1] Kenneth J. Arrow. *Social Choice and Individual Values*. Wiley, New York, 2nd edition, 1963.

[2] Vincent Conitzer and Tuomas Sandholm. Complexity of mechanism design. In Adnan Darwiche and Nir Friedman, editors, *Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference*, pages 103–110. Morgan Kaufmann, San Francisco, 2002.

[3] Vincent Conitzer and Tuomas Sandholm. Applications of automated mechanism design. Unpublished manuscript, available from http://www-2.cs.cmu.edu/∼conitzer/, 2003.

[4] Anthony Jameson, Stephan Baldes, and Thomas Kleinbauer. Enhancing mutual awareness in group recommender systems. In Bamshad Mobasher and Sarabjot S. Anand, editors, *Proceedings of the IJCAI 2003 Workshop on Intelligent Techniques for Web Personalization*. AAAI, Menlo Park, CA, 2003. Available from http://maya.cs.depaul.edu/~mobasher/itwp03/schedule.html.

[5] Louis Kaplow. Horizontal equity: New measures, unclear principles. Technical Report 7649, National Bureau of Economic Research, Cambridge, MA, 2000.

[6] Joseph F. McCarthy and Theodore D. Anagnost. MusicFX: An arbiter of group preferences for computer supported collaborative workouts. In *Proceedings of the 1998 Conference on Computer-Supported Cooperative Work*, pages 363–372, 1998.

[7] Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.

[8] Jeffrey S. Rosenschein and Gilad Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers*. MIT Press, Cambridge, MA, 1994.

[9] Yoav Shoham and Moshe Tenenholtz. Fair imposition. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1083–1088. Morgan Kaufmann, San Francisco, CA, 2001.

[10] Hal Varian. Mechanism design for computerized agents. In *USENIX Workshop on Electronic Commerce*, 1995.