Creating a Consensus Ranking of Proposals from Reviewers' Partial Ordinal Rankings

Wade D. Cook[†]

Boaz Golany[*]

Michal Penn[*]

Tal Raviv[‡]

[†]Schulich School of Business

York University

Toronto, Ontario M3J 1P3 Canada

[*]Faculty of Industrial Engineering and Management

Technion—Israel Institute of Technology

Haifa 32000, Israel

[‡]Sauder School of Business

University of British Columbia

Vancouver, BC V6T 1Z2 Canada

November 2004

Revised March 2005

Abstract

Peer review of research proposals and articles is an essential element in R&D processes world-wide. In most cases, each reviewer evaluates a small subset of the candidate proposals. The review board is then faced with the challenge of creating an overall "consensus" ranking on the basis of many partial rankings. In this paper we propose a branch and bound model to support the construction of an aggregate ranking from the partial rankings provided by the reviewers. In a recent paper we proposed ways to allocate proposals to reviewers so as to achieve the maximum possible overlap among the subsets of proposals allocated to different reviewers. Here, we develop a special branch and bound algorithm that utilizes the overlap generated through our earlier methods to enable discrimination in ranking the competing proposals. The effectiveness and efficiency of the algorithm is demonstrated with small numerical examples and tested through an extensive simulation experiment.

Keywords: Peer review, branch & bound algorithms.

1 Introduction

Peer review of research proposals and articles is an essential element in R&D processes and the academic community world-wide. Surprisingly, the operational issues involved in running a peer review process have drawn almost no attention from the operations research/management science community. The few articles that touch this area do not attempt to make the process work more effectively. Rather, they typically highlight the potential deficiencies of the process, suggest alternative quantitative models to replace it and compare their results to the outcomes of the peer review (e.g., Cardus et al. 1982, Hall et al. 1992).

In a recent paper (Cook et al., 2004) we argue that ordinal ranking of proposals is a valid (and often more practical than cardinal ranking) technique to evaluate proposals and developed methods to assign proposals to specific referees. In addition to the obvious need to match the reviewers' qualifications to the proposals, our methods lead to the maximal possible overlap in the subsets of proposals assigned to different referees. This overlap is essential when the reviewers are asked to provide their evaluations through pairwise ordinal rankings since otherwise, any overall ranking would be arbitrary.

In this paper, we address the problem of combining the judgements of the reviewers in a fair, objective and efficient manner. Over the past several decades various authors have examined the problem of combining individual preferences to form a compromise or consensus ranking. The manner in which preferences over the objects to be ranked (proposals, in our particular case) are expressed, depends on the level of possible quantification. In some situations cardinal or quantitative data on each of various attributes of the objects can be specified. In many practical applications, however, it is not possible to explicitly quantify the utility or value in a full cardinal format, and one must settle for the less specific ordinal specification. In some situations one can specify a complete "ranking" of the n objects on an ordinal scale in vector format $A = (a_1, a_2, \ldots, a_n)$, where $a_i \epsilon \{1, 2, \ldots, n\}$ is the rank position occupied by object i.

When such a ranking A^k is supplied by each member k of a committee of K members, one can then define a consensus of opinions in terms of the median ranking as discussed in Cook and Seiford (1978). From a practical point of view, if n is large, a full ranking may prove difficult, and the most that one can expect is to obtain partial rankings of subsets of the objects.

A common format for expressing preferences is to use 'pair-wise' comparisons. This mode of expression forces one to make a direct choice of one object over another when comparing two objects rather than requiring one to compare *all* objects simultaneously. As discussed in Cook and Kress (1991), consumer preferences over a set of products are commonly elicited from respondents by way of pair-wise comparisons. The consumer is generally asked questions of the form 'in terms of flavor, do you prefer product a, or product b?'. In many sporting events, such as round robin tournaments, outcomes from the matches are, by definition, reported as pair-wise comparisons (e.g., a defeated b). Thus, pair-wise comparisons provide a practical framework, in a wide variety of settings, for collecting information on ordinal preferences. It is particularly attractive when a comparison of all the objects is not possible and only a partial ranking may be supplied. In the case that an individual voter or committee member can only express preferences concerning a proper subset of the objects, then a partial ranking is the most information that this person can provide. In such a situation, vector representations as discussed above, make little practical sense, and one must then default to pair-wise comparisons. A pair-wise method is clearly advantageous in the current application where an individual reviewer will be asked to appraise only a subset of the proposals.

The problem of deriving a consensus among a set of ordinal preferences is one that arises in a wide variety of settings. As discussed above, much of market research is aimed at arriving at a consensus or compromise of opinions among a set of consumer preferences. Both sophisticated and ad hoc methods have been developed over time for deriving such a consensus. In a number of fields such as computer science, articles submitted to major conferences are refereed prior to acceptance for inclusion in the program. The submissions are sent to many reviewers, who return their evaluations to the program committee which then meets for a number of days of deliberations to

finalize the list of accepted papers. Reviewers generally do not treat these submissions in the same way as they would treat a journal submission (that is, they are not willing to invest the time and effort to do a full-fledged review). Hence, it is reasonable to ask each reviewer to rank order a limited number of submissions, and then aggregate the outcomes. Since the rankings serve only as raw-material to the committee (who may overrule the order recommended by some reviewers), ordinal ranking, and specifically, the use of pair-wise comparisons is a highly desirable approach. Finally, another interesting application suggested by Beg and Ahmad (2003) involves rank aggregation of partial rankings obtained through search engines on the World Wide Web. The above examples serve to illustrate the wide applicability of ordinal ranking and pairwise comparisons in practical settings.

The problem of deriving a *consensus ranking* from preferences provided in pairwise format was first examined by Kemeny and Snell (1962) and later by Bogart (1975) who extended the structure to partial orders. In neither case were possible solution methods presented. The problem of consensus ranking in the case that preferences are represented in vector (rank order) format has been investigated extensively by many researchers including Cook and Seiford (1978), Kirkwood and Sarin (1985) and Cook and Kress (1991), and various solution methods based on distance functions have been studied. The consensus ranking problem has also been approached from the point of view of various outranking methods such as that due to Roubens (1982). Further, a somewhat related problem is the tournament ranking problem as studied in Ali et al. (1986), Cook and Kress (1990), and Golany and Kress (1993).

Regardless of the representation used to elicit preferences, whether in vector or pair-wise comparison format, one of the most commonly used criteria for developing a compromise or consensus ranking is to minimize the number of "violations" (generally called the *minimum violations* consensus ranking). The idea is to obtain an overall ranking that displays the least number of cases where the opinions of the voters or respondents are *violated*. E.g., if the voter prefers a to b, yet the consensus ranking calls for b preferred to a, then a violation has occurred. While there are other criteria for deriving a consensus such as the Spearman foot rule distance technique, it is the

minimum violations method that is the most widely used in practice. For example, the overall ranking of players in a round-robin tournament (see Ali et al. (1986), and Cook and Kress (1990)) is intended to be one which deviates from the actual competition outcomes to the least extent possible. Most of the practical tools for aggregating consumer preferences are based on this idea. In the case, for example, where preferences are specified in ranking-vector format, it is common to compute the sum or average of ranks (across the set of consumer responses). The object with the lowest sum or average is ranked in first place, and so on. This is the well known Kendall scores method (Kendall (1962)), or the 'method of marks' due to Borda (1781). Cook and Seiford (1982) show that the average (hence sum) of ranks is equivalent to the minimum violations ranking in the ℓ_2 norm. Thus, even the 'ad hoc' techniques, such as a sum of ranks, is based on the idea of minimum violations.

In this paper we develop a branch and bound algorithm for deriving a consensus ranking of the proposals with minimum violations. We note that the minimum violation problem is NP-hard due to its equivalence to the minimum feedback arc set problem (see, e.g., Isaak and Narayan, 2004). Thus, using a branch and bound procedure is a reasonable approach. While it may be necessary to determine only a winning proposal, rather than a complete ranking of all the alternatives, the distance-based method that we use requires (in general) that a complete consensus ranking be constructed before a 'top ranked' proposal is actually found. As will be illustrated, a 'winning' proposal may emerge before the algorithm reaches the complete ranking of all the alternatives.

The rest of the paper is organized as follows. In §2 we present an algorithm to aggregate the partial matrices containing the pairwise evaluations of the reviewers into a consensus ranking and illustrate the implementation of the algorithm through a numerical example. In §3, we report on a large set of numerical examples used to evaluate the procedure proposed in the previous section. Section 4 concludes the paper.

2 Ranking Procedures

We start by formalizing some of the concepts that were already discussed in §1. A *complete ranking* of a set of N proposals $P = \{p_1, p_2, \ldots p_N\}$ is a permutation $R = (p_{j_1}, p_{j_2}, \ldots p_{j_N})$ of P. A *sub-ranking* of R is an ordered set of some consecutive proposals in R with the same order as in R. A *suffix* (*prefix*) of a ranking R is a sub-ranking of R that contains the last (first) proposal of R or an empty ranking. A *partial ranking* of the set P is an ordered subset of P.

Our objective is to develop a method to derive a complete ranking based on partial rankings provided by K reviewers. As described earlier, each reviewer will be assigned a subset of the N proposals to evaluate, and is expected to provide a partial ranking corresponding to this subset. For purposes herein, we assume that the preferences for reviewer k are given by a binary pairwise comparison structure through the *ranking matrix* $A^k = (a^k_{pq})$, where

$$a^k_{pq} = \begin{cases} 1 & \text{if proposal p is preferred to q} \\ -1 & \text{if proposal q is preferred to p} \\ 0 & \text{if p and q are not compared.} \end{cases}$$

Since it is senseless to compare any proposal p to itself, we set $a^k_{pp} = 0$ for all p and k. We assume here that each reviewer expresses a clear preference of one proposal over another when comparing two proposals. That is, we are assuming herein that tying two proposals that are being evaluated is not a valid option for the reviewer.

Kemeny and Snell (1962) prove that in the presence of a natural set of axioms, the unique distance function on the space of rankings is the absolute value functional. Specifically, the distance d between any two ranking matrices $A = (a_{pq})$ and $B = (b_{pq})$ can be given by

$$d(A, B) = \frac{1}{2} \sum_{p=1}^{N} \sum_{q=1}^{N} |a_{pq} - b_{pq}|.$$

Definition 2.1 *Given a collection of partial rankings* $\{A^1, A^2, \ldots, A^K\}$ *the* ***consen-***

sus value *of a ranking B (in matrix representation) is given by*

$$\mathcal{M}(B) = \sum_{k=1}^{K} d(A^k, B) = \frac{1}{2} \sum_{k=1}^{K} \sum_{p=1}^{N} \sum_{q=1}^{N} |a_{pq}^k - b_{pq}|. \tag{1}$$

In the sequel, we simplify notation by using the expression $\mathcal{M}(R)$ where the argument R is a ranking (in vector representation) and not the matrix it induces.

Definition 2.2 *A ranking R^* is an* ***optimal consensus ranking*** *if it minimizes the consensus value $\mathcal{M}(R)$ over all possible rankings R.*

The pairwise comparison consensus ranking problem has, to the best of the authors' knowledge, never been viewed previously from a strictly mathematical programming perspective. The principal difficulty in deriving a mathematical programming structure has to do with the requirement that the elements b_{pq} in the corresponding ranking matrix B satisfy requisite transitivity conditions.

One can express the problem as an integer programming formulation as follows. Define a set of binary variables x_{pq} for all $p \neq q$ where $x_{pq} = 1$ if proposal p is preferred to proposal q and is 0 otherwise. Also, for each pair of proposals $\{p, q\}$, let the summary statistics r_{pq} be the number of reviewers who preferred q to p. Hence, r_{pq} represents the number of violations that will occur if p is ranked ahead of q in the final ranking. Now, solve the binary integer programming problem.

$$\max \sum_{\{p,q\} \in P^2 : p \neq q} r_{pq} x_{pq} \tag{2}$$

s.t.

$$\begin{aligned} x_{pq} + x_{qs} &\leq 1 + x_{ps} \qquad \forall \{p, q, s\} \in P^3 : p \neq q, p \neq s, q \neq s \\ x_{pq} + x_{qp} &= 1 \qquad \forall \{p, q\} \in P^2 \\ x_{pq} &\in \{0, 1\} \end{aligned}$$

It is important to point out that while one can theoretically derive a consensus ranking by solving the above problem, size becomes a major issue. Specifically, the number of constraints is given by $N(N-1)(N-2)+0.5N(N-1)$, which for the case of, say, 60 proposals yields approximately $207,090$ constraints. For this reason we propose to solve this problem using a branch and bound algorithm. Again, we point out that a number of procedures for the various ranking representations are discussed in Cook and Kress (1991). However, no formal methodologies with accompanying computational tests have been presented for the Kemeny and Snell (1962) consensus method.

In order to initiate a branch and bound algorithm we define u_k as the number of proposals which reviewer k evaluates. We note that the values of r_{pq} and u_k are all the information needed to calculate the consensuses value $\mathcal{M}(R)$ for any ranking R.

$$\mathcal{M}(R) = \sum_{\{p,q\}\in P: p \succ_R q} r_{pq} + \frac{1}{2}\sum_{k=1}^{K} \binom{N-u_k}{2}. \tag{3}$$

Where $p \succ_R q$ denotes the fact that p precedes q in the ranking R. The first summation in (3) counts the number of cases where a reviewer prefers p over q but q is ranked in a higher position in the ranking R. Each such case contributes 1 to $\mathcal{M}(R)$. The second summation counts the cases where a reviewer expresses no preference between p and q (since he did not review both of these proposals). Each such case contributes $\frac{1}{2}$ to the consensus value. We note that the second summation is uniquely defined by the values of N and $u_1,\ldots,u_k$ which remain constant for all rankings. Therefore, a ranking that minimizes

$$M(R) = \sum_{\{p,q\}\in P: p \succ_R q} r_{pq}. \tag{4}$$

also minimizes $\mathcal{M}(R)$. Hence, from here on we shall consider the minimization of $M(R)$ instead of $\mathcal{M}(R)$.

Proposition 2.1 (Separability property) *Consider an optimal consensus ranking R^* of a set of proposals P. Let $R_1,\ldots,R_m$ be a partition of R^* into m consecutive*

sub-rankings which divide P into $P_1, \ldots, P_m$. Then $R_1, \ldots, R_m$ are optimal consensus rankings of $P_1, \ldots, P_m$ with respect to the same reviewers' preferences.

Proof. Consider the value of the ranking R^*,

$$M(R^*) = \sum_{\{p,q\} \in P : p \succ_R q} r_{pq} = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{p \in P_i, q \in P_j} r_{pq} + \sum_{i=1}^{m} \sum_{\{p,q\} \in P_i : p \succ_R q} r_{pq} \ . \qquad (5)$$

Now, assume by contradiction that for some i there exists an R_i' (a ranking of the proposals of P_i) such that $M(R_i') < M(R_i)$. Replacing R_i by R_i' affects only the i^{th} term in the second summation in (5) and so the optimality of R^* is contradicted. ■

Definition 2.3 *[Eligible Prefix (EP)]: A prefix $R = (q_1, ..., q_s)$ is said to be eligible if the following conditions hold*

1. $\sum_{q \in P \setminus R} r_{q_s,q} \leq \sum_{q \in P \setminus R} r_{q,q_s}$.
2. $\sum_{i=a}^{s-1} r_{q_s,q_i} \geq \sum_{i=a}^{s-1} r_{q_i,q_s}$ *for all* $a = 1, ..., s-1$.
3. $\sum_{i=a}^{s-1} r_{q_i,q_a} \geq \sum_{i=a}^{s-1} r_{q_a,q_i}$ *for all* $a = 1, ..., s-1$.
4. $\sum_{i=a}^{s-1} r_{q_s,q_i} > \sum_{i=a}^{s-1} r_{q_i,q_s}$ *for all* $a = 1, ..., s-1$ *such that* $q_a > q_s$.

Proposition 2.2 *Any prefix of an optimal ranking admits conditions 1–3 of Definition 2.3*

Proof. Consider a ranking $R = (q_1, ..., q_s, ..., q_n)$. If condition 1 is violated for the prefix $(q_1, ..., q_s)$ then the ranking $(q_1, ..., q_{s-1}, q_{s+1}, ..., q_n, q_s)$ is better than R. To see this, note that by moving q_s to the end, the consensus measure is increased by $\sum_{q \in P \setminus R} r_{q,q_s}$ because of the reviewers that prefer q_s over the proposals $\{q_{s+1}, \ldots, q_n\}$ but it decreases by $\sum_{q \in P \setminus R} r_{q_s,q}$ because of the reviewers that prefer these proposals over q_s.

For similar considerations, if condition 2 is violated then the ranking $(q_1, ..., q_{a-1},$- $q_s, q_a, ..., q_{s-1}, q_{s+1}, ..., q_n)$ is better than R. If the third condition is violated then the ranking $(q_1, ..., q_{a-1}, q_{a+1}, ..., q_{s-1}, q_a, q_s, ..., q_n)$ is better. ■

Proposition 2.3 *An optimal ranking that admits condition 4 for all its prefixes exists.*

Proof. Consider an optimal ranking $R_1 = (q_1, ..., q_n)$ in which the prefix $(q_1, ..., q_s)$ is the shortest sub-ranking that demonstrates violation of condition 4 for some $a' < s$ and let a be the minimal such index with respect to s. Note that since R is an optimal ranking then condition 2 holds and so the inequality of condition 4 is violated with equality. That is, $\sum_{i=a}^{s-1} r_{q_s,q_i} = \sum_{i=a}^{s-1} r_{q_i,q_s}$. Now, by moving q_s to the location before q_a we create a new optimal ranking $R_2 = (q_1, ..., q_{a-1}, q_s, q_a, ..., q_n)$. If the ranking R_2 still violates condition 4 we can apply the same procedure again and so on. Since the number of possible ranking (and in particular optimal ranking) is finite there are only two possibilities to consider. Either this procedure is ended in R_k that admits condition 4 or this is a cyclic procedure and so for some k we have $R_k = R_1$. Assume by contradiction that the procedure may be cyclic. Let b be the "highest" location in the ranking that is changed during the cycle $R_1, ..., R_k$. That is any proposal that is ranked above q_b in R_1 preserves it location throughout the cycle. Let us review the sequence of proposals that are ranked as highest throughout the cycle. Note that any such proposal must admit lower index than it predecessor in the b^{th} location. Otherwise, it can not violate condition 4. Hence the procedure can not be cyclic and we are done. ■

In the algorithm below we start with an empty partial ranking R. For each proposal p which has not yet been ranked and is an eligible immediate successor of R, we construct a lower bound. If this lower bound is lower than the value of the best known solution, we append proposal p to the partial ranking and store this ranking as a new node in our branch and bound tree. Next, we run a probing heuristic, described below, to extend the obtained prefix into a complete ranking in order to construct an upper bound. If this upper bound is lower than the value of the currently known best solution, we take it as a new incumbent best known solution.

A *lower bound* on all the rankings with prefix R is given by,

$$\underline{M}(R) = \sum_{\{p,q\}\subseteq P_1 : p \succ_R q} r_{pq} + \sum_{p\in P_1, q\in P_2} r_{pq} + \sum_{\{p,q\}\subseteq P_2} \min\{r_{pq}, r_{qp}\} \tag{6}$$

where P_1 is the set of proposals ranked by R and P_2 is the complementary set of the other proposals. We also note that if R' is a ranking created by appending a single proposal p to a ranking R (at the bottom) then

$$\underline{M}(R') = \underline{M}(R) + \sum_{q\in P_2} \{r_{pq} - \min\{r_{pq}, r_{qp}\}\} \ . \tag{7}$$

Thus, updating the lower bound after extending a given partial ranking is easier than calculating it from scratch.

An *upper bound* can be calculated by extending the partial ranking at each node using some fast heuristic method. Below we describe how a majority rule is iteratively applied to carry out this task.

A Probing Heuristic

Input: a set of proposals P, with reviewers preferences summery statistics $\{r_{pq}\}$.
Output: A complete ranking with R as prefix and the resultant consensus measure.

Initialization Let P be the set of proposals not in R.

Iteration While $P \neq \emptyset$, find a proposal $p \in P$ with minimum ratio $\frac{\sum_{q\in P1} r_{pq}}{\sum_{q\in P} r_{pq} + \sum_{q\in P} r_{qp}}$.
Add p to R as the last proposal, remove it from P and repeat.

Output Return the ranking R.

We note that this procedure can be replaced by any sound heuristic including neighborhood search heuristics. However since the procedure is to be employed in any node inserted to our branching tree it should be a quick one.

We are now ready to present the main algorithm to generate an optimal consensus ranking.

Main Algorithm - Optimal Consensus Ranking

Input: Set of proposals P and summary statistics of reviewer decisions $\{r_{pq}\}$ for all $\{p, q\} \subseteq P$.

Step 0 (Initialization): Find dominating and dominated partial rankings with respect to the set of all proposals. Set these proposals as top and bottom proposals, T and B, respectively and remove them from P. Let R_0 be the initial empty ranking. Use (6) to calculate its lower bound $\underline{M}(R_0)$ and store it as the root node of the branch and bound tree. Define this node as an active one. Use the probing heuristic to obtain an initial best known ranking and store its consensus measure as upper bound Y. If the lower bound of the root node equals this upper bound go to Step 3.

Step 1 (Selecting the node to branch from): Choose from the set of active nodes of the branch and bound tree the one with the lowest lower bound. In case of ties, select the node of the longest partial ranking. Denote the ranking of the selected node as R_c. Deactivate the selected node. If its lower bound $\underline{M}(R_c)$ is lower than the current best known solution go to Step 3.

Step 2: For all eligible immediate successors of R_c, $p \in S(R_c)$:

Step 2a(Branching): Construct a candidate partial ranking R_n with R_c followed by p.

Step 2b (Bounding): Use (7) to calculate a lower bound $\underline{M}(R_n)$. If this lower bound is not lower than the currently best known solution go to Step 3.

Step2c (Create new node): Add R_n as an active node to the tree and store $\underline{M}(R_n)$ with this node.

Step2d (Probing): Use the probing heuristic to extend this partial ranking into a complete one R_n^e. Use (7) [or (4)] to calculate the consensus measure $M(R_n^e)$. If this measure is lower than the value of the currently best known

solution, store it as the new incumbent best known solution and deactivate all nodes whose lower bounds are greater than the new upper bound.

Step 3 (Termination) : If there are still active nodes, return to Step 1. Otherwise, stop and return the incumbent ranking preceded by T and followed by B.

It is important to note that one may encounter alternate optima when applying the algorithm provided herein. In such instances, it may make sense to incorporate a secondary objective function that would select from the alternative optimal solutions, one which is "the most balanced" in terms of distributing the violations as uniformly as possible across the reviewers. In this way, the result would be seen by reviewers as being a fair representation of their opinions. Such a mechanism might be a meaningful way of resolving ties in optimal solutions to the problem. However, implementing such a mechanism requires a major algorithmic undertaking. Hence, in the current paper we have not attempted to address this issue. This may be the subject of future research.

2.1 A Numerical Example

Consider an example with $N = 6$ proposals, and $K = 5$ reviewers:

Reviewer	*Proposals*	*Ranking*
1	$\{1, 2, 3, 5\}$	$1 \succ 3 \succ 2 \succ 5$
2	$\{1, 2, 4, 6\}$	$2 \succ 1 \succ 4 \succ 6$
3	$\{3, 4, 5, 6\}$	$4 \succ 3 \succ 5 \succ 6$
4	$\{1, 4, 5, 6\}$	$6 \succ 1 \succ 4 \succ 5$
5	$\{1, 2, 5, 6\}$	$6 \succ 2 \succ 3 \succ 1$

The table below summarizes the data for the proposal pairs:

$$r_{pq} = \begin{pmatrix} 0 & 2 & 1 & 0 & 0 & 2 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 2 & 1 & 0 & 0 & 0 & 1 \\ 2 & 1 & 2 & 2 & 0 & 1 \\ 1 & 1 & 1 & 2 & 1 & 0 \end{pmatrix}$$

Algorithm results

Initialization: We start by identifying dominating and dominated partial rankings. Note that for all $p \in P = \{1, \ldots, 6\}$ we have $r_{2p} \leq r_{p2}$ and so we can set $T = (2)$ and remove $\{2\}$ from P. Next, we identify proposal 5 as a dominated one and so we set $B = (5)$ and $P = \{1, 3, 4, 6\}$. With the new P, proposal 3 is also dominated and so we add it to B ($B = (3, 5)$) and remove it from P ($P = \{1, 4, 6\}$). At this point, no dominating or dominated proposals are left. For example, proposal 1 is not a dominating proposal with respect to P since $r_{16} > r_{61}$ and it is not a dominated one since $r_{14} < r_{41}$. The remaining reviewers' evaluation matrix is now given by:

$$r_{pq} = \begin{array}{c|ccc} & (1) & (4) & (6) \\ (1) & 0 & 0 & 2 \\ (4) & 2 & 0 & 1 \\ (6) & 1 & 2 & 0 \end{array}$$

Next, we calculate a lower bound for all the rankings (of the remaining proposals) using (6) and obtain $\underline{M}(\textit{empty ranking}) = 2$. To obtain an upper bound, we use the probing heuristic. We first calculate majority indices $M_1 = \frac{2}{2+3} = 0.4$, $M_4 = \frac{2}{2+3} = 0.4$, $M_6 = \frac{3}{3+3} = 0.5$. We obtain the ranking $1 \succ 4 \succ 6$ and store it as incumbent solution. Using (4) we calculate the consensus measure of this ranking $M(1 \succ 4 \succ 6) = 2+1 = 3$. This is the upper bound for now.

Branching: We process the root node with the empty ranking. Since all the proposals in $P = \{1, 4, 6\}$ are eligible successors of this ranking, we check them all.

- Adding proposal 1: We are left with $P = \{4, 6\}$ where proposal 4 dominates proposal 6 and so we obtain the ranking $1 \succ 4 \succ 6$ with $M(1 \succ 4 \succ 6) = 2+1 = 3$. This is a complete ranking which is not better than our current optimal solution so there is no need to add it to the branch and bound tree.
- Adding proposal 4: We are left with $P = \{1, 6\}$ where proposal 6 dominates proposal 1 and so we obtain the ranking $4 \succ 6 \succ 1$ with $M(4 \succ 6 \succ 1) = 4$. This is a complete ranking which is not better than our current optimal solution so there is no need to add it to the branch and bound tree.
- Adding proposal 6: We are left with $P = \{1, 4\}$ where proposal 4 dominates proposal 1 and so we obtain the ranking $6 \succ 4 \succ 1$ with $M(6 \succ 4 \succ 1) = 5$. This is a complete ranking which is again not better than our current optimal solution.

We are now left with no active nodes in our branch and bound tree and so we declare the current best known solution as the optimal ranking of $P = \{1, 4, 6\}$. Finally, we "merge" the dominating and dominated rankings, obtained at the initialization step, to this ranking to obtain the optimal solution $2 \succ 1 \succ 4 \succ 6 \succ 3 \succ 5$.

3 Numerical Experiments

Here we demonstrate the applicability of the ranking algorithm presented in the previous section. Our testing platform was a Pentium 4, 2Ghz with 512Mb RAM that run under Windows XP. We coded the algorithm in C++ with the aid of LEDA (see Mehlhorn and Näher, 1999) and compiled it with Microsoft Visual C++ 6.0. We constructed five classes, each with 25 test problems, with different numbers of proposals and pairwise reviewers as shown in Table 1.

Class	# of proposals	# of pairwise comparisons
A	20	253
B	20	316
C	30	380
D	40	360
E	60	3150

Table 1: Five classes of test problems

For each of the 125 problems described in Table 1 we created 3 different sets of reviewer decision matrices using the following procedure. A so called "objective grade" was drawn from a normal distribution $N(75, 10)$ for each proposal. For all the proposals that are actually checked by each reviewer (according to the heuristic solution obtained at the previous subsection) we generated a reviewer grade which is the sum of the objective grade and a normally distributed noise $N(0, \sigma^2)$ with $\sigma = 1, 4$ or 9. The ranking of each reviewer (given as input to our algorithm) was constructed based on these grades. The C++ code and our test problems with detailed solutions can be downloaded from "http://iew3.technion.ac.il/Home/Users/golany/Download". In Table 2 we present the average and worst case running time (in seconds) for each of the five classes and three levels of noise. In addition we present the percentage of the problems that could be solved within the time limit of 1800 seconds. Recall that for each class we have 25 instances, so we solved $25 \times 4 \times 3 = 300$ problems.

The table demonstrates the fact that we are able to obtain an optimal ranking for most problem instances with up to 40 proposals.

It is apparent from the table that the level of noise adversely affects the processing time. However, any process of ranking is based on the belief that the reviewers are capable of delivering nearly objective rankings. Moreover, we note that a consistent bias of a reviewer (e.g., a tendency to assess all proposals as better than what they "really" are) does not affect our procedure. This is because our algorithm uses only relative ranking as input. This is a fundamental advantage of our procedure as compared to some traditional procedures where the reviewers are asked to quote an absolute grade

	$\sigma = 1$		$\sigma = 4$		$\sigma = 9$	
Class	Avg. (max) running time	% solved to optimality	Avg. (max) running time	% solved to optimality	Avg. (max) running time	% solved to optimality
A	< 0.01 (< 0.01)	100	< 0.01 (0.02)	100	0.02 (0.06)	100
B	< 0.01 (< 0.01)	100	< 0.01 (0.02)	100	0.02 (0.9)	100
C	< 0.01 (0.02)	100	0.27 (1.52)	100	6.48 (29.33)	100
D	0.01 (0.11)	100	4.68 (36)	100	332 (1800)	96
E	0.08 (0.20)	100	1714 (3600)	72	-	-

Table 2: Running time of the ranking algorithm and percentage of the problems solved within 1800 seconds (3600 seconds for class E).

for each proposal and the ranking is based on the average grade of the proposals.

The fact that classes A and B take a similar amount of time to process (both with 20 proposals and with 50 and 60 reviewers, respectively) implies that our algorithm is insensitive to the number of reviewers. This is not surprising since the heuristic works on a summary statistics of the reviewers responses' rather than on the actual responses. In fact, we expect that large number of reviewers will reduce the inconsistency of the rankings obtained by majority rule and hence will make the problem easier to solve.

In order to evaluate the contribution of our probing heuristic to the performance of the algorithm, a version of the algorithm that uses a naive upper bound was created. Here, the prefix in each node is arbitrarily extended by the remaining proposals instead of using the majority rule. We present the results of this experiment in the table below.

By comparing Table 2 with Table 3, one can observe that the probing heuristic does reduce the running time. The effect is particulary dramatic for the larger instances with low or medium noise levels where the probing heuristic is likely to catch optimal solutions at high levels of the branching tree.

We compared our method with the performance of the commercial Integer Programming Solver MOSEK using the formulation we presented above. It turns out that MOSEK was capable of solving all of our test problems but with significantly larger

	$\sigma = 1$		$\sigma = 4$		$\sigma = 9$	
Class	Avg. (max) running time	% solved to optimality	Avg. (max) running time	% solved to optimality	Avg. (max) running time	% solved to optimality
A	< 0.01 (0.02)	100	0.01 (0.08)	100	0.03 (0.11)	100
B	< 0.01 (0.02)	100	0.01 (0.09)	100	0.03 (0.11)	100
C	0.07 (1.08)	100	1.78 (12.95)	100	8.94 (28.19)	100
D	1.16 (23)	100	26 (573)	100	444 (1800)	88

Table 3: Running time of the ranking algorithm with naive probing heuristic and percentage of the problems solved within 1800 seconds.

running times for the low noise instances. For example the 60 proposals problems of class E with low noise were solved in 678 seconds on average and the moderate noise instances ($\sigma = 4$) were solved in over 1000 seconds on average.

4 Conclusions

The process of reviewing, evaluating and finally ranking research or research-related manuscripts (e.g., submissions to academic competitions, research proposals) is an integral part of academia. This process is based on peer review by researchers who usually perform this task on a voluntary basis. In many cases the submissions are numerous and diverse in their subject topics and therefore require a large and diversified group of reviewers or judges. Given the reviewers' partial evaluations in pairwise-comparison format, the problem addressed in this paper is how to produce a fair and robust aggregate ranking of the submissions.

The pairwise-comparison representation of preferences is common in a wide range of practical applications, where opinions from respondents (voters, reviewers, consumers) can be obtained only in an ordinal format. In the current application, where each reviewer sees only a subset of proposals, hence only a partial ordering can be provided by each, this preference representation format is ideal. The theoretical issue of deriving a consensus of reviewers' opinions, as expressed by the supplied partial pairwise

comparisons, has been studied extensively in the literature, dating back to the original work of Kemeny and Snell (1962). Despite the wide applicability of the consensus idea, and the accompanying criterion of minimizing the number of violations (in regard to reviewers' preferences),however, little effort has been placed on the actual development of effective algorithms for finding such a consensus. The principal difficulty lies in the requirement that the matrix of final preferences must be transitive.

The current paper develops the requisite theoretical basis for deriving a minimum violations consensus ranking. We present a branch and bound algorithm for computing a consensus among a set voter responses, and demonstrate its solution capability for a range of problem sizes, in terms of the numbers of reviewers and proposals. This provides the user with a clear indication of the types of real world problems that can be solved using this methodology. The algorithm and accompanying software described herein are, thus, important tools for solving large scale consensus ranking applications.

The methodology developed herein applies to complete or strong ranking structures only. An important area for further research is that of deriving a consensus ranking allowing for tied preferences. This problem of weak rankings will be the subject of future investigation.

Acknowledgment: The authors wish to thank an anonymous referee who suggested a variation of integer programming formulation (2).

References

Ali, I., W.D. Cook, M. Kress. 1986. On the minimum violations ranking of a tournament, *Management Science*, 32(6), 660–672.

Beg, M.M.S., N. Ahmed. 2003. Soft computing techniques for rank aggregation on the world wide web. *World Wide Web: Internet and Web Information Systems*, 6, 5–22.

Bogart, K.P. 1975. Preference structures II: distances between asymmetric relations, *SIAM Journal of Applied Math.*, 29(2), 254–262.

Borda, J.C. 1781. Memoir sur les elections au scrutin. *Histoire de l'Academie Royale des Sciences.*

Cardus, D., M.J. Fuhrer, A.W. Martin, R.M. Thrall. 1982. Use of benefit-cost-analysis in the peer-review of proposed research, *Management Science*, 28(4), 439–445.

Cook, W.D., B. Golany, M. Kress, M. Penn, T. Raviv. 2005. Optimal Allocation of Proposals to Reviewers to Facilitate Effective Ranking, *Management Science*, forthcoming.

Cook, W.D., M. Kress. 1990. An mth generation model for weak ranking of players in a tournament, *Journal of the Operational Research Society*, 41(12), 1111-1119.

Cook, W.D., M. Kress. 1991. *Ordinal information and preference structures: decision models and applications*, Prentice-Hall, New Jersey.

Cook, W.D, L.M. Seiford. 1978. Priority ranking and consensus formation, *Management Science*, 24(16), 1721-1732.

Cook, W.D, L.M. Seiford. 1982. The Borda-Kendall consensus method for priority ranking problems, *Management Science*, 28(6), 621–637.

Golany, B., M. Kress. 1993. A multi-criteria evaluation of methods for obtaining weights from ratio-scale matrices, *European Journal of Operational Research*, 69(2), 210–220.

Hall, N.G., J.C. Hershey, L.G. Kessler, R.C. Stotts. 1992. A model for making project funding decisions at the national cancer institute, *Operations Research*, 40(6), 1040–1052.

Isaak, G., D.A. Narayan. 2004. A classification of tournaments having an acyclic tournament as a minimum feedback arc set, *Information Processing Letters*, 92(3), 107–111.

Kemeny, J.G., L.J. Snell. 1962. Preference ranking: an axiomatic approach, in *Mathematical Models in the Social Science*, Ginn, Boston, 9–23.

Kendall, M. 1962. *Rank Correlation Methods*. 3rd Edition, Hafner, New York.

Kirkwood, C.W., R.K. Sarin. 1985. Ranking with partial information—a method and an application, *Operations Research*, 33(1), 38–48.

Mehlhorn, K., S. Näher. 1999. Leda—a platform for combinatorial optimization and geometric computing, *Cambridge University Press*

Roubens, M. 1982. Preference relations on actions and criteria in multi-criteria decision making, *European Journal of Operational Research*, 10, 51–55.