Forming a ranking from tied evaluations: a case of an online, interactive student peer assessment system

Lihi Dery

Abstract

In higher education courses, peer assessment activities are common for keeping students engaged during presentations. Defining precisely how students assess the work of others requires careful consideration. Asking the student for numeric grades is the most common method. However, students tend to assign high grades to most projects. Aggregating peer assessments, therefore, results in all projects receiving the same grade. Moreover, students might strategically assign low grades to the projects of others so that their projects will shine. Asking students to order all projects from best to worst imposes a high cognitive load on them, as studies have shown that people find it difficult to order more than a handful of items. To address these issues, we propose a novel peer rating model, *R2R*, consisting of (a) an algorithm that elicits student assessments and (b) a protocol for aggregating grades to produce a single order. The algorithm asks students to evaluate projects and answer pairwise comparison queries. These are then aggregated into a ranking over the projects. *R2R* was deployed and tested in a university course and showed promising results, including fewer ties between alternatives and a significant reduction in the communication load on students.

1 Introduction

In the context of forming ranked lists over alternatives, two important problems are encountered: *Preference elicitation* and *Preference aggregation* [3, 4, 5, 14, 20]. For instance, when creating a list of competitors ordered by their performance scores (e.g., athletes or performers) [9, 12, 23], it is necessary first to elicit and then to aggregate preferences to form a ranking. Similarly, when creating a list of students ordered by their perceived academic performance, the preferences of peers or teachers must be elicited and aggregated.

In peer grading or peer assessment systems, where students are required to grade their peers [32, 33], preference elicitation encounters an additional challenge: students tend to be very generous in the grades they assign to their peers. This issue is illustrated in Figure 1, which shows the grades that 34 students gave to 10 projects their peers presented in class during a session in a university course. The median score for all projects in this session was either "Excellent" or "Very good". This is not an irregularity. We collected peer grading data from six different class sessions. All exhibited similar performances. Other studies observe the same trend [21]: students are laxer, and teachers are more strict in their reviews. This does not mean that students think all the projects are excellent. As we show later, when prompted, students can state which projects they prefer over others. However, a simple preference elicitation grading system may fail to capture these preferences.

In peer assessment systems, preference aggregation is also problematic. Simply computing the average rating is not a good solution, as it is easily manipulable [30]. Although this is true in many scenarios, in peer assessment systems, the voters are also the candidates (or friends of the candidates), so the risk is higher. Consider, for example, three student projects: c_1, c_2, and c_3. If the average rating is used, a voter who favors c_1 will assign it the highest possible score, but to ensure that c_1 ends up in the first place, the voter might

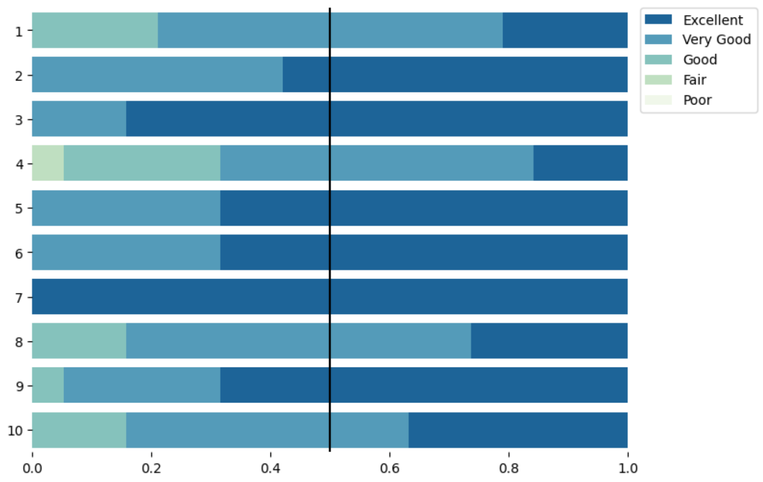


Figure 1: The distribution of grades of ten projects in a university course. The y-axis shows the project numbers and the x-axis is the cumulative percentage of students who gave these projects a particular grade, as specified in the legend. The vertical black line indicates the median score: it is either "Excellent" or "Very good".

also assign c_2 and c_3 the lowest possible score. Computing the median rating offers a less-manipulable solution but may result in many ties between the alternatives. The Majority judgment voting protocol [2] addresses the problem of tie-breaking the median, and recently some more median tie-breaking methods have been suggested by Fabre [10]. Nevertheless, these methods still result in ties when the input is student peer rating grades.

One potential approach to address the rating aggregation problem involves modifying the preference elicitation protocol. Rather than permitting agents to assign ratings to alternatives, an alternative strategy could involve requesting agents to provide a ranked preference list that encompasses all alternatives or to assign distinct scores to each alternative (which effectively translates into ranking the alternatives). Indeed, some peer review systems have suggested collecting ordinal preferences [27]. However, experts and non-experts alike are limited in the number of preferences they can meaningfully order — sometimes they simply do not hold a rank over the alternatives (see chapter 15 in [2]), and oftentimes they find it hard to rank more than seven items from best to worst [17]. Pairwise comparison queries such as "do you prefer this alternative over that alternative?" may assist people in forming a ranking [15, 8]. However, this can be perceived as a tiresome task for voters. For example, the required number of comparison queries for ten alternatives is 45.

In summary, limiting voters' task to rating alternatives conceals important information about their preferences and results in a problem of preference aggregation. Limiting the voters' task to ranking alternatives requires a cognitive and communication effort that may burden the voters.

In this paper, we wish to benefit from both cardinal preferences, expressed by ratings (or grades), and ordinal preferences, as expressed with pairwise comparisons. We propose a novel peer assessment model, $R2R$, consisting of (a) an algorithm that elicits student assessments and (b) a protocol for aggregating grades into a single order. The algorithm asks students to evaluate projects and answer pairwise comparison queries. These are then aggregated into a ranking over the projects. **Our contributions:**

- **Rate and then compare** - The $R2R$ algorithm elicits voter preferences while minimizing the cognitive and communication overload. Voters are asked to rate alternatives and respond to pairwise comparison queries only when necessary.

- **Improved tie-breaking for the median** - we present a protocol for aggregating the voters' preferences into a single ranking. Our protocol is based on the median and Copeland's voting rule. The protocol reduces the number of ties in the final aggregated ranking.
- **Personal ranking made easy** - Our method always produces a personal ranking of the alternatives for each voter, with no ties. This may be beneficial for a voter trying to figure out her preferences. For example, an employee (or a researcher) views candidates (or presentations) and would like to contact them in descending order and offer them a job (or a research position).

Some research (e.g. [30]) focuses on strategic manipulation in conference and journal peer reviews, however, this is not the focus of this paper. This study focuses on peer review in an academic classroom. Peer review increases motivation, cultivates critical thinking, and increases engagement in the classroom (see e.g. [34, 16, 28]). Many peer review systems aggregate the grades using either the mean or the median (e.g. [24, 26, 35]). The reliability and validity of classroom peer review have been extensively researched (e.g. [16, 18]). Some systems attempt to address problems in these aggregation systems by assigning different weights to each reviewer according to their credibility (e.g., [31]). Others attempt to statistically correct bias [29], control who reviews whom, or incorporate AI-based methods [7].

In our *R2R* system, all reviewers are equal, and all reviewers rate all projects. Possible bias in reviews is mitigated by employing a common grading language as done by Balinski and Laraki [2] and by using ordinal as well as cardinal reviews. We did not find signs of other possible bias (see Section 4).

2 Preliminaries

Let there be n voters and m candidates: $V = \{v_1, v_2, \ldots, v_n\}, C = \{c_1, c_2, \ldots, c_m\}$. Each voter v_i assigns a score s to each candidate. Let s_i^j denote the score assigned by voter v_i to candidate c_j. The score is assigned from a predefined domain of discrete values $D = \{d_{min} \ldots d_{max}\}$ where d_{min} and d_{max} are the lowest and highest values respectively ($d_{min} \leq s_i^j \leq d_{max}$).

The ordered set of all scores assigned to candidate c_j is S^j. The median is a candidate's middle-most score when n is odd, and the $n/2$ middle-most score when n is even. As illustrated in Figure 1, the median score of most candidates is “Excellent” or “Very Good” (these scores translate into the numerical scores of “5” and “4”). Therefore, candidates cannot be ranked solely according to the median, and a tie-breaking mechanism is required.

Existing median-based voting rules are presented in section 2.1. As our rule is based on the median and Copeland voting, Copeland voting is presented in section 2.2.

2.1 Median based voting rules

A few mechanisms exist for tie-breaking the median between candidates. They are all based on the same idea: first, compute the candidate's median. Then add a quantity based on what Fabre [10] terms as proponents (p) and opponents(q). The mechanisms differ in their use of p and q. The proponent p is the share of scores higher than the median α: $p_j = \sum_{i|s_i^j > \alpha} s_i^j$. The opponent q is the share of scores lower than the median α: $q_j = \sum_{i|s_i^j < \alpha} s_i^j$.

The score of a candidate is the sum of its median grade α and a tie-breaking rule. Fabre [10] defines three rules: Typical judgment, Central judgment, and Usual judgment.

Definition 2.1 (Typical judgement)**.** Typical judgment is based on the difference between non-median groups: $c^{\Delta} = \alpha + p - q$.

Definition 2.2 (Central judgement)**.** Central judgment is based on the relative share of proponents: $c^{\sigma} = \alpha + 0.5 \cdot \frac{p-q}{p+q}$.

Definition 2.3 (Usual judgement)**.** Usual judgment is based on the normalized difference between non-median groups: $c^{v} = \alpha + 0.5 \cdot \frac{p-q}{1-p-q}$.

Fabre [10] shows that majority judgment [2] can also be defined using just α, p and q :

Definition 2.4 (Majority judgement)**.** $c^{mj} = \alpha + \mathbb{1}_{p>q} - \mathbb{1}_{p\leq q}$. [1] If ties remain, the median of the tied candidates is dropped, and then mj is recomputed for the tied candidates. This procedure is repeated until all ties are resolved. Note that for ties Fabre [10] suggest an alternative method that results in the same output.

Example 1. Assume the set of scores assigned to candidate c_j by ten voters is:

$$S^j = \{5, 5, 5, 4, 4, 4, 3, 3, 3, 2\}$$

The median is, therefore: $\alpha = 4$. Three voters assigned a score which is above the median, and four voters assigned a score below the median. Therefore, the proponent and opponent are, respectively: $p = \frac{3}{10}, q = \frac{4}{10}$.

The score of c_j according to Typical, Central, Usual, and Majority judgments, is:

$$c_j^{\Delta} = 4 + \frac{3}{10} - q = \frac{4}{10} = 3.9$$

$$c_j^{\sigma} = 4 + 0.5 \cdot \frac{-0.1}{0.7} \approx 3.92$$

$$c_j^{v} = 4 + 0.5 \cdot \frac{-0.1}{1.1} \approx 3.95$$

$$c_j^{mj} = 4 - 1 = 3$$

Candidate scores can be computed using the rules above. A ranking over the candidates can be obtained by simply ordering them according to their scores.

2.2 Copeland voting

In section 2.1, only the scores given to the candidates are used. Herein, we also use the voters' preferences over the candidates. We write $c_i \succ c_j$ when c_i is preferred over c_j. Let $N(c_i, c_j)$ denote the number of voters who prefer $c_i \succ c_j$.

Copeland [6] suggested a voting system that is based on pairwise comparisons. The method finds a Condorcet winner when such a winner exists. Each candidate is compared to every other candidate. The candidate obtains one point when it is preferred over another candidate by the majority of the voters. If two candidate tie, they both receive half a point. Adding up the points results is the Copeland score of each candidate.

[1] $\mathbb{1}_{f(x)}$ is an indicator function of $f(x)$. For example, if $p > q$ then $\mathbb{1}_{p>q} = 1$

A family of Copeland methods, *Copeland*$^\alpha$, was suggested by Faliszewski et al. [11]. The difference is in tie-breaking the alternatives. The fraction of points each candidate receives in case of a tie can be set to any rational number: $0 \leq \alpha \leq 1$. The Copeland score of a candidate is thus the number of points it obtained plus the number of ties times α.[2]

Formally:

Definition 2.5 (*Copeland*$^\alpha$ voting rule)**.** The score of a candidate is the sum of victories in pairwise comparisons:
$c^{copeland^\alpha} = \sum_{j=1,\forall j\neq i}^{m} [N(c_i, c_j) > n/2] + \alpha \cdot \sum_{j=1,\forall j\neq i}^{m} [N(c_i, c_j) \equiv N(c_j, c_i)]$

3 The model

Voters are usually asked to either rank or rate alternatives. However, we claim that reducing their task to just this or the other conceals important information about their full preferences. We propose a model consisting of two parts:

- *R2R* **Elicitation** - an algorithm that elicits voter preferences while minimizing the cognitive and communication overload. Voters are asked to rate alternatives. Voters are asked to respond to pairwise comparison queries when necessary.
- *R2R* **Aggregation** - a protocol for aggregating the voters' preferences into a single ranking.

We herein describe both parts.

3.1 R2R elicitation

We adopted a grading scale consisting of five grades ($D = 1, 2 \ldots 5$), based on the research of Miller [17] which suggested that people can distinguish between seven plus or minus two levels of intensity. To minimize bias in the ratings, we utilized a standardized grading language proposed by Balinski and Laraki [2] that maps to the following five grades: Excellent (5), Very good (4), Good (3), Fair (2), and Poor (1). Voters were instructed to evaluate each project holistically and in accordance with the specified project requirements. A project graded as "Excellent" was one that met the requirements in an exceptional manner. These grades can be mapped to a Rating set.

Definition 3.1 (Rating Set)**.** A rating set T is a tuple $< c, s >$ containing candidates (c) and their scores (s). The rating set of voter v_i is denoted $\tau_i \in T$.

When the voters assign a different score to each candidate, constructing the preference profile is straightforward. For example, if $s_i^2 > s_i^1 > s_i^3$ then the preference profile π_i of voter v_i is: $\{(c_2, 1), (c_1, 2), (c_3, 3)\}$ meaning $c_2 \succ c_1 \succ c_3$.

Definition 3.2 (Preference profile)**.** A preference profile $\pi \in \Pi$ is a tuple $< c, p >$ containing candidates (c) and their ranked position (p). The set of all possible preference profiles (the permutation space) is Π. We write $c_i \succ c_j$ when c_i is preferred over c_j. For $k < l$ the preference profile is thus: $\{\ldots (c_i, p_k), (c_j, p_l) \ldots\}$. The preference profile of voter v_i is π_i.

However, if v_i assigns the same score to two or more of the candidates, we executed pairwise comparison queries $q \in Q$ to determine which of the two candidates v_i prefers. We

[2]The Copeland α is not the same as the median α. To be consistent with previous research (and thus make it easier for readers familiar with the literature), we denote them both as α, but mention in the text which α we are referring to.

assume that a voter can always respond to a query so that a pairwise comparison query $q(v_i, c_j, c_k)$ has only two possible responses: $c_j \succ c_k$ or $c_k \succ c_j$.

Here, we deviate from the standard literature, where it is assumed that each voter holds either a rating set or a preference profile, and define a ranked rating set. A ranked rating set contains attributes of both a rating set and a preference profile, thus allowing us to save more information about the voters' preferences.

Definition 3.3 (Ranked Rating Set)**.** A ranked rating set $\psi \in \Psi$ is a tuple $< c, d, p >$ containing candidates (c), their scores (s) and their position (p). As in the ranking set, the position is determined by the ranking. The rating set of voter v_i is denoted $\psi_i \in \Psi$.

We present a method, Rating to Ranking, $R2R$ which builds a ranked rating set while eliciting the needed information from the voters. A pseudo-code is provided in Algorithm 1 followed by a detailed description and a running example.

Algorithm 1 Rating to Ranking (R2R) elicitation

```
1: Input:
       a set of alternatives: $c_1, c_2 \ldots, c_M$
       a set of voters: $v_1, v_2 \ldots, v_N$
       a score domain $D = \{d_{min} \ldots d_{max}\}$
2: for $v_i \in V$ do
3:     $\psi_i \in \Psi \leftarrow []$
4:     for $c_j \in C$ do
5:         voter declares $s_i^j \in D$
6:         $count \leftarrow count(\psi_i, s_i^j)$
7:         if $count \leq 1$ then
8:             $insert(\psi_i, c_j)$
9:         else
10:            $p \leftarrow index(\psi_i, s_i^j)$
11:            while $(count \geq 1)$ do
12:                $c_p \leftarrow candAtIndex(\psi_i, p)$
13:                execute query $q(v_i, c_j, c_p)$
14:                if $c_j \succ c_p$ then
15:                    $p - -$
16:                    $count - -$
17:                else
18:                    exit while loop
19:            $insert(\psi_i, c_j, p)$
20: Output:
       $\psi_i$ - a ranked rating set for each voter $v_i$
```

The algorithm receives as input a set of M alternatives, N voters, and a score domain D. Each voter is sequentially asked to provide scores for all candidates (lines 2-5). Note that this process can also be done the other way around: for each candidate, all voters are requested to submit their scores. In either case, voter v_i assigns a score s_i^j to candidate c_j (line 5). Subsequently, the algorithm counts how many times this particular score (s_i^j) has been previously submitted by the voter (line 6). If the voter has not submitted this score before, the score is inserted to ψ (lines 7-8) while ensuring its ordered placement (lines 7-8), utilizing the *insert* function. In case the score s_i^j already exists within ψ, the algorithm retrieves the index of the last position where it is found (line 10) and identifies the candidate c_p at that position (line 12). A pairwise query is then executed to determine the relative ranking between the two candidates (line 13). If c_j is ranked higher than c_p ($c_j \succ c_p$) the algorithm proceeds to the next position and repeats the query process (lines 14-16, followed by line 11). Otherwise, if c_j is ranked lower than c_p ($c_p \succ c_j$), the algorithm proceeds to insert c_j at the identified position p (lines 19). Finally, the output is a ranked rating set (line 20).

It is important to note that this pairwise comparison approach prevents the occurrence of Condorcet cycles.

Example 2. Consider one voter v_1 and four candidates: c_1, c_2, c_3, c_4. Scores are given in domain $D = 1, 2 \dots 5$. Assume that the voter's rating set is: $\tau_1 = \{(c_1, 5), (c_2, 4), (c_3, 5), (c_4, 5)\}$ and the voter's preference profile is: $\pi_1 = \{c_3 \succ c_4 \succ c_1 \succ c_2\}$. The rating set and the preference profile are initially unknown. The goal is to determine the ranked rating set.

At first, v_1 declares her score for c_1: $s_1^1 = 5$. Since this is the first score declared, the ranked rating set is updated to $\psi_1 = \{(c_1, 5, 0)\}$. Then, the next score is declared: $s_1^2 = 4$. Since ψ does not contain this score, ψ_1 is updated without any queries: $\psi_1 = \{(c_1, 5, 0), (c_2, 4, 1)\}$. The next score declared is $s_1^3 = 5$. Since this score is already in ψ_1, a query is issued: $q(v_1, c_1, c_3)$. The voter responds by stating: $c_3 \succ c_1$. Thus c_3 is now added and ψ_1 positions of candidates is updated: $\psi_1 = \{(\mathbf{c_3}, \mathbf{5}, \mathbf{0}), (c_1, 5, \mathbf{1}), (c_2, 4, \mathbf{2})\}$ (changes to ψ_1 are marked in bold). Lastly, v_1 declares her score for c_4: $s_1^4 = 5$. This causes a query to be issued: $q(v_1, c_1, c_4)$. The voter responds with: $c_4 \succ c_1$, so yet another query is issued: $q(v_1, c_3, c_4)$. The voter responds with: $c_3 \succ c_4$ and ψ_1 is finalized: $\psi_1 = \{(c_3, 5, 0), (\mathbf{c_4}, \mathbf{5}, \mathbf{1}), (c_1, 5, \mathbf{2}), (c_2, 4, \mathbf{3})\}$ (again, changes to ψ_1 are marked in bold).

3.2 R2R aggregation

The output of the *R2R* elicitation phase is a set of ranking sets Ψ. First, the median α is computed from the scores (s) in Ψ. Then, alternatives are ordered according to their median. When two or more alternatives have the same median, some pairwise comparison method is used to determine their order. Herein, we use $Copeland^\alpha$. The output is an aggregated preference profile. This is a collective preference profile of all of the voters. A pseudo-code is provided in Algorithm 2.

Algorithm 2 Rating to Ranking (R2R) aggregation

```
1: Input:
       a set of ranked rating profiles $\psi_i \in \Psi$
2: $medians \leftarrow []$
3: for $c_i \in C$ do
4:     $medians \leftarrow median(c_i)$
5: order alternatives by median
6: for $c \in$ tied medians do
7:     order by $Copeland^\alpha$ score
8: Output:
       An aggregated preference profile $\pi$ (a collective preference profile)
```

4 User Study

In order to collect ranked rating sets, we implemented a peer-rating system according to our *R2R* model.[3]

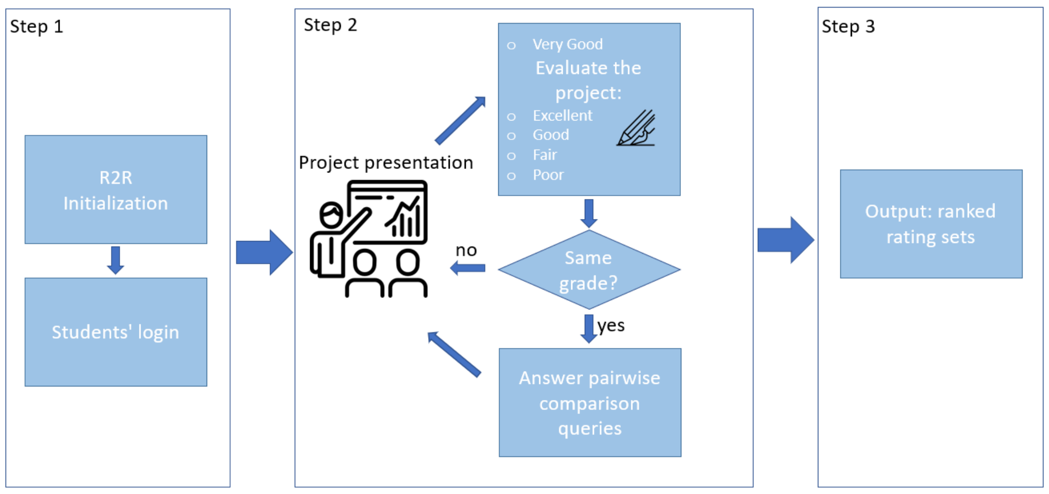


Figure 2: R2R system flow (the icons are by Pause08 from www.flaticon.com.)

The main steps of the system are described in Figure 2. To initialize the system, project names and numbers were fed into the system beforehand by a system admin. A link to the system was then distributed to the students, who logged in with a username and password (step 1).

During the presentation sessions (step 2), we employed the *R2R* system to facilitate project evaluation. After each presentation, students were asked to use the system to assess the quality of the project. Thus, they were prompted to submit one of five evaluations: Excellent, Very good, Good, Fair, and Poor. Although students had the option to provide written comments, these comments were not used in the study. In the event that the same grade was assigned by a student to more than one project, we executed pairwise comparison queries following algorithm 1.

In step 3, the system outputs ranked rating sets, one set for each participating student. Screenshots from the system, illustrating step 3, can be found in the Appendix.

The system was deployed as part of a peer grading process in a university course. Course students presented their final projects in class. The other students present were instructed to insert evaluations into the *R2R* system. Each student participated in one class presentation session. In each session, 9-12 projects were evaluated. We held six different sessions (sessions $a - f$). The study received ethical approval from the institutional review board of the university. Session dates and project names are concealed to maintain the participants' privacy.

In total, 222 students participated in the experiments. However, 28 students failed to rate all the projects in their session. Another 31 students did not report their preferences according to the order the projects were presented in class. This led us to suspect these students did not pay attention and reported arbitrary preferences. These students were removed from the analysis. We were thus left with a total of 163 students whose responses were included in the data analysis.

[3]The code is readily available at `https://github.com/Matan-Lange/ratings-to-rankings/tree/hub_version`. A link to the online system will be sent to interested parties upon request.

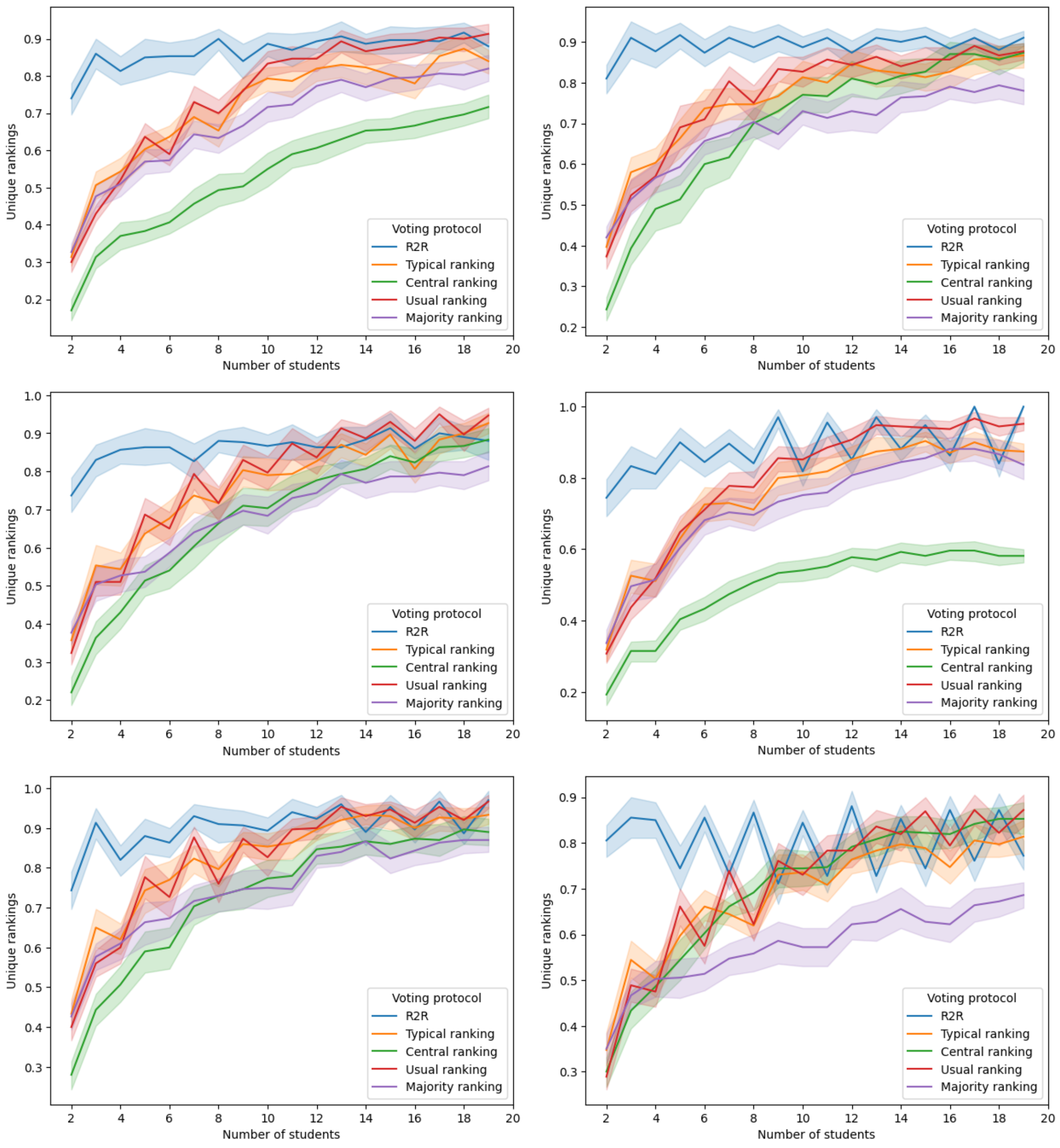


Figure 3: Unique rankings in sessions a to f (top to bottom, left to right).

Table 1: Data collection statistics: for each of the six sessions, we present the number of participants, number of projects, number of projects with a median grade of excellent\very good\good and the average number of pairwise comparison queries each participant received (standard deviation is in the parentheses).

Session	Number of participants	Projects	Number of median grades (excellent, very good, good, fair, poor)	Pairwise comparison queries
a	27	10	6,4,0,0,0	13.9(5.7)
b	24	10	5,4,1,0,0	10.8(2.7)
c	29	10	4,6,0,0,0	13.2(54.9)
d	22	9	2,7,0,0,0	11.8(3.9)
e	27	10	3,5,2,0,0	10.6(2.5)
f	34	12	0,5,6,1,0	14.6(5.9)

Communication overload: A summary of the participation data is presented in Table 1. Most projects received a median grade of either "excellent" or "very good". For example, in session a (first row in Table 1), 34 students were asked to rate ten projects. Six projects received the median grade of "excellent", four received the median grade of "very good" and no projects received the median grade of "good".

For 9,10 and 12 projects, the maximum amount of pairwise queries is 36, 45 and 66, respectively. In the six sessions, the average number of issued pairwise queries lay in the range of 10.87-14.71 (last column in Table 1). Thus, $R2R$ reduces the communication load by $69\%, 75\%, 71\%, 67\%, 75\%, 77\%$ in sessions a, b, c, d, e, f respectively. In other words, on average, students had to respond to less than 30% of the total number of possible pairwise queries.

Ties in rankings: We compared the proposed $R2R$ method to existing methods: Majority ranking [2], Typical ranking, Central ranking, and Usual ranking. The three latter are the rankings retrieved from the Typical judgment, Central judgment, and Usual judgment methods [10].

The unique number of rankings in each method was computed, which refers to the number of projects that were not tied in their ranking. The experiment was conducted with a varying number of voters (students in the user study), ranging from 2 to 20, which were sampled without replacement. Each experiment ran 30 times. The α in $Copeland^{\alpha}$ was set to $\alpha = \frac{1}{3}$, as it provided better results than $\alpha = \{0, \frac{1}{2}, 1\}$. Setting α to values smaller than $\frac{1}{3}$ did not yield an improvement.

The results show that as the number of voters increased, the methods exhibited better performance with fewer ties and more unique rankings. Tie-breaking is easier when the number of voters is odd. This explains why the increase is not smooth. In all sessions, $R2R$ outperformed the other methods when the number of voters is small. The other methods, especially Usual Ranking, were more competitive when there are more voters. This suggests that $R2R$ is especially useful when the number of voters is relatively small. Figure 3 illustrates these comparisons on a. Due to space constraints, results for all of the sessions are provided in the appendix. All sessions display a similar trend.

Order bias: We examined whether the presentation order affected the users' answers. We hypothesized that perhaps students assign high scores to the first projects presented, and then after they experience a few pairwise comparison prompts, they begin to distribute their scores better (perhaps in order to avoid the prompts). However, Figure 4 assured us this is not the case. The y-axis is the percentage of voters that assigned a score of "very good" (left figure) and "excellent" (right figure). The x-axis is the vote order, from 1 to 12

Table 2: Primacy and recency bias proportions.

Evaluation	Descending order	Acsending order
Excellent	0.19	0.16
Very good	0.17	0.1
Good	0.19	0.3
Fair	0.32	0.42
Pair	0.14	0.43

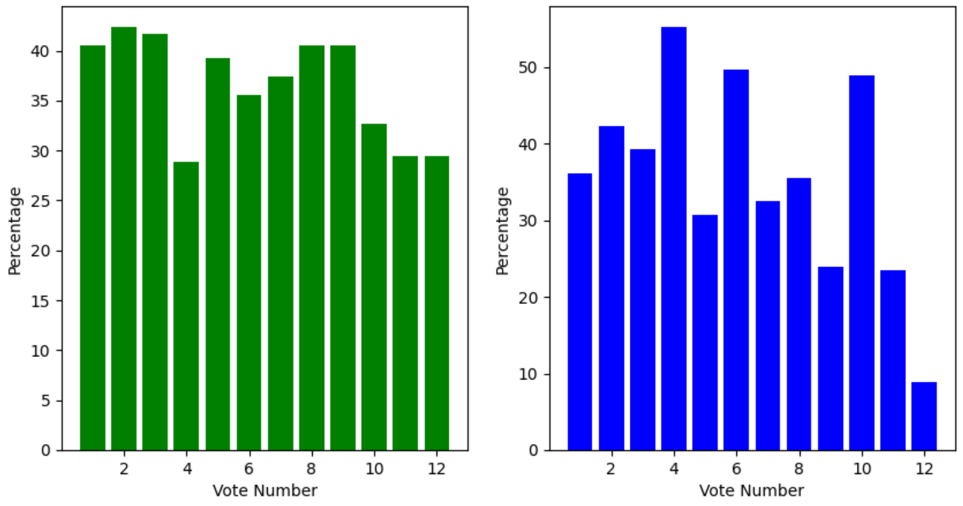


Figure 4: Percentage of voters that assigned a score of “very good” (left) and “excellent” (right).

(when there are 12 projects). We can see that there is no skewness to the left. Therefore, we cannot detect a tendency to begin with high scores and decrease them.

We next examined whether users are prone to a primacy or recency bias – when prompted with a pairwise query, do users tend to select the first or last projects they saw? We examined all the projects in a session that received the same score and then computed the percentage of cases projects were ordered in ascending or descending order. For example, consider a case where the order of presentations was: $c_1, c_2 \ldots c_{10}$. A student gave an “excellent” to projects: c_2, c_3, c_8 and the preference profile is: $c_2 \succ c_3 \succ c_8$. We then say that the student exhibits a primacy bias. If a student assigned a “very good” to projects: c_1, c_5 and has a preference profile: $c_5 \succ c_1$ we say this is a recency bias. A student can have a recency bias for some evaluations and a primacy for others. The results are reported in Table 2. We did not detect a trend, but this is an initial result and a deeper analysis should be performed here. We note that the primacy-recency bias has been studied in the context of memory [19] and decisions such as what link to click on [13]. However, we did not find research on primacy-recency effects in preference elicitation.

5 Discussion

We presented *R2R*, a peer assessment model that converts ratings to rankings. The model is designed to elicit voter preferences by asking them to evaluate alternatives using cardinal preferences, such as scores or grades. If necessary, voters are also asked to make pairwise comparisons between two alternatives to provide ordinal preferences. The model then aggregates these preferences to output a ranked list of alternatives.

In our experiments, we found that *R2R* outperformed existing methods by reducing the number of ties in the output, especially for small groups of voters (up to 10-15). Additionally, *R2R* imposes a lower communication load on the user compared to pairwise comparison methods, as voters only need to evaluate each alternative once, instead of repeatedly comparing pairs of alternatives. Moreover, the cognitive load is lower than in models that require users to rank all alternatives, as they only need to assign a score or grade to each alternative. Overall, *R2R* provides a simple and effective approach for ranking alternatives in various settings, such as peer assessment in academic settings or performance evaluation in workplaces.

The *R2R* system was implemented and utilized in our classroom, but the elicitation and aggregation methods presented in this paper are applicable to other tasks as well. In the workplace, for instance, group or team leaders often need to evaluate the performance of their employees, as noted by [1]. This is a sensitive task since an employee's position on the final performance list usually determines the bonus they will receive. Managers can use our model to establish their personal ranked list of employees.

At present, voters cannot express indifference between alternatives in our system. While this guarantees that the output does not include ties, it also limits the decision-maker's preferences. It remains to be seen what effect removing this limitation would have.

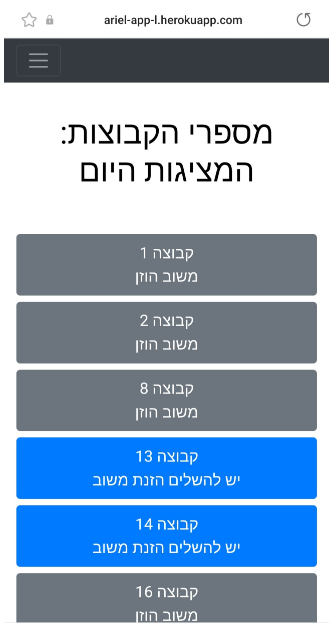
One possible avenue for future research is to explore the incorporation of multiple pairwise comparison types, as suggested by [22]. Another potential direction is to integrate a truth serum, as recommended by [25], which would incentivize voters to provide their honest preferences, as they would be evaluated. These research directions could further enhance the usefulness and applicability of the *R2R* system in various domains.

Acknowledgments

We would like to extend our special thanks to Matan Lange for his implementation of the *R2R* system and his valuable assistance during the user study. We also appreciate the assistance of Koby Karady in the data analysis of order bias.

Appendix

Figures 5-7 present screenshots from the *R*2*R* system as seen on a smartphone. The system is in Hebrew; an English version is currently being developed. Figure 3 displays the performance of *R*2*R* in comparison to other aggregation methods across all six sessions (a to f).



ariel-app-l.herokuapp.com

מספרי הקבוצות:
המציגות היום

קבוצה 1
משוב הוזן

קבוצה 2
משוב הוזן

קבוצה 8
משוב הוזן

קבוצה 13
יש להשלים הזנת משוב

קבוצה 14
יש להשלים הזנת משוב

קבוצה 16
משוב הוזן

Figure 5: The main screen a student views after logging on to the *R*2*R* system using a smartphone. Title: "Projects presented today". The blue and grey boxes contain the project's number with the text: "evaluation completed" (grey boxes) or "evaluation needed" (blue boxes). More projects will appear when the student scrolls down her phone screen.

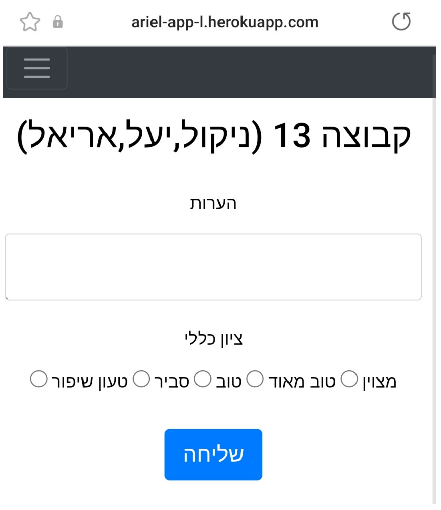
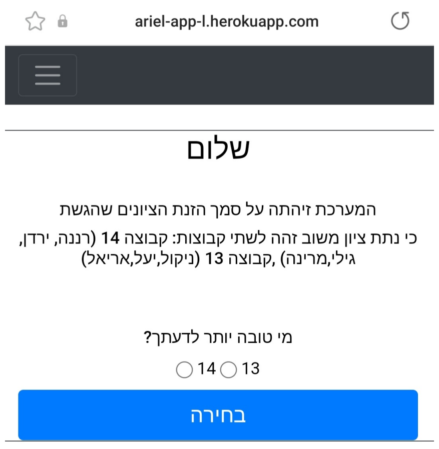

ariel-app-l.herokuapp.com
קבוצה 13 (ניקול,יעל,אריאל)
הערות
ציון כללי
מצוין ◯ טוב מאוד ◯ טוב ◯ סביר ◯ טעון שיפור ◯
שליחה

Figure 6: The screen a student sees when required to evaluate a specific project. Top line: “Group 13 (name of project owners)”. Second line: “Comments”. Third line: “General grade”. The fourth line contains five radio buttons: “Excellent”, “Very good”, “Good”, “Fair”, “Poor”



המערכת זיהתה על סמך הזנת הציונים שהגשת

כי נתת ציון משוב זהה לשתי קבוצות: קבוצה 14 (רננה, ירדן, גילי,מרינה), קבוצה 13 (ניקול,יעל,אריאל)

מי טובה יותר לדעתך?

Figure 7: *R2R* Pairwise comparison question displayed to the student when she assigns the same evaluation to two projects. Translation to English: “The system has detected that you gave two projects the same grade: project 14 (name of project owners) and project 13 (name of projects owners). Which project do you think is better?” The student has two radion buttons to choose from, one for project 13 and one for project 14.

References

[1] Richard D Arvey and Kevin R Murphy. Performance evaluation in work settings. *Annual review of psychology*, 49(1):141–168, 1998.

[2] Michel Balinski and Rida Laraki. *Majority judgment: measuring, ranking, and electing.* MIT press, 2011.

[3] Dorothea Baumeister and Jörg Rothe. Preference aggregation by voting. *Economics and computation: An introduction to algorithmic game theory, computational social choice, and fair division*, pages 197–325, 2016.

[4] Gleb Beliakov, Ana Pradera, Tomasa Calvo, et al. *Aggregation functions: A guide for practitioners*, volume 221. Springer, 2007.

[5] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of computational social choice.* Cambridge University Press, 2016.

[6] Arthur H Copeland. A reasonable social welfare function. In *Mimeographed notes from a Seminar on Applications of Mathematics to the Social Sciences, University of Michigan*, 1951.

[7] Ali Darvishi, Hassan Khosravi, Shazia Sadiq, and Dragan Gašević. Incorporating ai and learning analytics to build trustworthy peer assessment systems. *British Journal of Educational Technology*, 2022.

[8] Lihi Naamani Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reaching a joint decision with minimal elicitation of voter preferences. *Information Sciences*, 278:466–487, 2014.

[9] M Pino Díaz-Pereira, Ivan Gomez-Conde, Merly Escalona, and David N Olivieri. Automatic recognition and scoring of olympic rhythmic gymnastic movements. *Human movement science*, 34:63–80, 2014.

[10] Adrien Fabre. Tie-breaking the highest median: alternatives to the majority judgment. *Social Choice and Welfare*, 56(1):101–124, 2021.

[11] Piotr Faliszewski, Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. Llull and copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, 35:275–341, 2009.

[12] Victor Ginsburgh and Abdul G Noury. The eurovision song contest. is voting political or cultural? *European Journal of Political Economy*, 24(1):41–52, 2008.

[13] Yukyung Lee and Carolyn A Lin. Exploring the serial position effects of online consumer reviews on heuristic vs. systematic information processing and consumer decision-making. *Journal of Internet Commerce*, 21(3):297–319, 2022.

[14] Ofrit Lesser, Lihi Naamani-Dery, Meir Kalech, and Yuval Elovici. Group decision support for leisure activities using voting and social networks. *Group Decision and Negotiation*, 26(3):473–494, 2017.

[15] Tyler Lu and Craig Boutilier. Learning mallows models with pairwise preferences. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 145–152, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

[16] Heng Luo, Anthony Robinson, and Jae-Young Park. Peer grading in a mooc: Reliability, validity, and perceived effects. *Online Learning Journal*, 18(2), 2014.

[17] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[18] Rafael Molina-Carmona, Rosana Satorre-Cuerda, Compañ-Rosique PATRICIA, and Faraón Llorens-Largo. Metrics for estimating validity, reliability and bias in peer assessment. *International Journal of Engineering Education*, 34(3), 2018.

[19] Jamie Murphy, Charles Hofacker, and Richard Mizerski. Primacy and recency effects on clicking behavior. *Journal of computer-mediated communication*, 11(2):522–535, 2006.

[20] Lihi Naamani-Dery, Meir Kalech, Lior Rokach, and Bracha Shapira. Reducing preference elicitation in group decision making. *Expert Systems with Applications*, 61: 246–261, 2016.

[21] Ali Mansoori Nejad and Omer Hassan Ali Mahfoodh. Assessment of oral presentations: Effectiveness of self-, peer-, and teacher assessments. *International Journal of Instruction*, 12(3):615–632, 2019.

[22] Mark EJ Newman. Ranking with multiple types of pairwise comparisons. *Proceedings of the Royal Society A*, 2022.

[23] António Osório. Performance evaluation: subjectivity, bias and judgment style in sport. *Group Decision and Negotiation*, 29:655–678, 2020.

[24] Dwayne E Paré and Steve Joordens. Peering into large lectures: examining peer and expert mark agreement using peerscholar, an online peer assessment tool. *Journal of Computer Assisted Learning*, 24(6):526–540, 2008.

[25] Drazen Prelec. A bayesian truth serum for subjective data. *science*, 306(5695):462–466, 2004.

[26] Helen Purchase and John Hamer. Peer-review in practice: eight years of aropä. *Assessment & Evaluation in Higher Education*, 43(7):1146–1165, 2018.

[27] Karthik Raman and Thorsten Joachims. Methods for ordinal peer grading. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1037–1046, 2014.

[28] Krishneel Reddy, Tony Harland, Rob Wass, and Nave Wald. Student peer review as a process of knowledge creation through dialogue. *Higher Education Research & Development*, 40(4):825–837, 2021.

[29] Juan Ramón Rico-Juan, Antonio-Javier Gallego, Jose J Valero-Mas, and Jorge Calvo-Zaragoza. Statistical semi-supervised system for grading multiple peer-reviewed open-ended works. *Computers & Education*, 126:264–282, 2018.

[30] Ivan Stelmakh, Nihar B Shah, and Aarti Singh. Catch me if i can: Detecting strategic behaviour in peer assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4794–4802, 2021.

[31] Dapeng Tao, Jun Cheng, Zhengtao Yu, Kun Yue, and Lizhen Wang. Domain-weighted majority voting for crowdsourcing. *IEEE transactions on neural networks and learning systems*, 30(1):163–174, 2018.

[32] Keith Topping. Peer assessment between students in colleges and universities. *Review of educational Research*, 68(3):249–276, 1998.

[33] Keith J Topping. Peer assessment. *Theory into practice*, 48(1):20–27, 2009.

[34] Thu Thuy Vu and Gloria Dall'Alba. Students' experience of peer assessment in a professional course. *Assessment & Evaluation in Higher Education*, 32(5):541–556, 2007.

[35] David Kofoed Wind, Rasmus Malthe Jørgensen, and Simon Lind Hansen. Peer feedback with peergrade. In *ICEL 2018 13th International Conference on e-Learning*, page 184. Academic Conferences and publishing limited, 2018.

Lihi Dery
Ariel University
Ariel, Israel
Email: `lihid@ariel.ac.il`