Partition with Two-Stages: What Works and How?

Anonymous submission

Abstract

Peer-evaluation systems are used when sets of agents evaluate each other in order to select the best k among them. These are commonly used in various real-world settings, such as in academic conferences, in which those submitting papers are also reviewing papers. Recently, conferences have attempted to focus their review resources by moving to a two-stage mechanism, in which some papers are eliminated after a first stage, and then more reviews are added to those which are left.

A major strategyproof mechanism that has been suggested for peer-evaluation is *Partition*, and we investigate how this mechanism performs when adapted to a two-stage system. Unlike what may be expected, we show that the two-stage mechanism's advantage is not for papers on the threshold of acceptance, but for those below the top-most papers, but not edge-case, ranking wise. We examine how the two-stage mechanism affects the quality of the selected agents, and what are the parameters that improve its performance – how many candidates to eliminate in the first stage, and how to divide the number of reviews between the stages. This is highly dependent on the number of required agents, the number of reviews requested from agents, and reviewers' correlation.

Introduction

Peer-evaluation, the process in which a group judges its members and selects the best, is a process humans have engaged with for thousands of years. In many ancient cases, selection by lot was done (sortition of sorts), as it was assumed there is a divine intervention (see Samuel I, chapter 10). Later, people tried to choose the best person for a job by various selection methods (e.g., the selection of the Pope from – and by – the College of Cardinals). While in small groups this can be akin to voting, the main difference from elections is that the set of candidates and voters are the same, and furthermore, the case of the selection of a single winner is less common in comparison to the desire to select a set of k best agents (or to find a ranking of all agents).

More recently, peer-evaluation is commonly practiced in many companies and organizations, for example, as part of employees' evaluation process or advancement protocol.Academics are intimately familiar with peer evaluation, as refereed academic conferences (such as most computer-science conferences, e.g., AAAI) implement such a system: those submitting papers are often tasked with evaluating other papers. Indeed, in the past few years many conferences have decreed that paper authors must be willing to review other papers.

Research in this field has mostly focused on *strategyproof* mechanisms, i.e., ways in which agents cannot improve their chances of being selected by giving an untruthful report on others. One of the initial mechanisms suggested for this was *Partition* (Alon et al. 2011), in which agents are divided into two groups, and each agent reviews agents which are not in their own group. These reviews ultimately create a ranking of each group, and the top $\frac{k}{2}$ is selected from each.

In parallel to these efforts, academic conferences have struggled to deal with a different, orthogonal problem: the problem of using a limited resource – the number of papers that can be reviewed by agents – with the desire to select the best set of papers. As research has shown (Lawrence 2022), reviewers are quite "noisy", i.e., not in agreement with each other on a ground-truth of the ranking of all papers. Therefore there is a desire to have more reviews for each paper, so as to increase the chance of getting a better signal regarding its quality. One approach to deal with this issue, initially used at the AAAI 2021 conference, and since then more widely in AI conferences, is two-stage reviewing (Leyton-Brown et al. 2022). In the first stage, each reviewer reviews only a few papers, and based on this relatively-noisy signal, papers which received very bad reviews are eliminated, leaving fewer papers, which reviewers' remaining reviews can be focused on. Thus, instead of all papers receiving the same amount of reviews, those eliminated in the first stage receive fewer, while the remaining ones get more, allowing, hopefully, for a better selection of the best papers.

In this paper we provide an analysis the Partition mechanism (which remains strategyproof in a two-stage setting), looking into two related questions:

1. What is the quality of a two-stage Partition mechanism? Does it improve over its single stage variant, and where do these improvements manifest themselves (i.e., which papers benefit from it)?
2. What parameters of two-stage Partition mechanism lead to better results? That is, if each reviewer reviews, overall, m agents, how many of those should be done in the first round and how many in the second round? Furthermore, how many papers should be eliminated in the first round (and we add an additional possibility – to further focus reviewer attention – to also select very good papers

in the first round, just as bad papers are eliminated)?

Using extensive simulations, we examine these questions in settings that vary the number of agents we wish to choose (k), the number of reviews from each agent (m), and how noisy agents' views are.

We will present two variants of a two-stage Partition mechanism, followed by an examination of how well the two-stage system improves over the single-stage variant. As Partition is the most widely-studied strategyproof peer-evaluation mechanism, we believe this can help start a more thorough examination of the effect of two-stage mechanisms on peer-evaluation methods. Partition also has the advantage of many other strategyproof mechanism building upon it, allowing a Partition analysis to go a long way to helping establish a basis for other mechanisms.

Contribution We show that two-stage Partition very significantly improves over regular Partition. While we expected this improvement to be in borderline agents (that is, those near the threshold of the top k papers), this turned out to be wrong – the main improvement was in the chance of the papers around best $k/2$ agents to get included. The advantage over Partition is stronger the fewer reviews required from each agent, the fewer agents are selected, and the noisier agents' views are. The different parameters of the two-stage mechanism are highly influenced by the k, m and the correlation of views, and we detail their connection.

Related Work

The AI community's attention to peer-evaluation was sparked by the NSF proposal, based on Merrifield and Saari (2009), to use peer-evaluation to divide telescope time. That mechanism incentivised agents not to report their true views of others, but instead, to report as their view what they think the consensus opinion will be. This undesirable property (encouraging misreporting) prodded researchers to suggest strategyproof mechanisms, in which agents are never better off by misreporting their true beliefs on others (Shah 2022).

One prominent mechanism, suggested in Alon et al. (2011), is Partition, in which agents are divided into two groups, with each group reviewing the other one, thus preserving strategyproofness (as each reviewer can only influence the ranking of agents in a group it is not a member of). In the original paper, as well as some further ones (Holzman and Moulin 2013), the setting considered was that of nomination, i.e., agents either approve or disapprove of a paper, so no ranking or more granular grades. Furthermore, some papers focused only on the selection of a single agent ($k = 1$) (Holzman and Moulin 2013; Fischer and Klimm 2015; Bousquet, Norin, and Vetta 2014), though a few expanded beyond that (Bjelde, Fischer, and Klimm 2017), including with real-world data (Xu et al. 2019). A variant of Partition with more than two clusters and different weights to each group, *Dollar Partition*, was suggested in Aziz et al. (2016), though it could return fewer than k agents, an issue fixed in *Exact Dollar Partition* (Aziz et al. 2019).

Beyond Partition, several other mechanisms have been suggested. Kurokawa et al. (2015) suggested *Credible Subset*, a mechanism based on identifying agents with a potential to manipulate, and including them in the set of potential winners, though that mechanism had a probability of selecting no agents at all. Gao, Wright, and Leyton-Brown (2019) suggested using a low-probability event of verifying reviewers, punishable very heavily to encourage agents to report truthfully. Mattei, Turrini, and Zhydkov (2020) proposed *PeerNomination*, which reverts to a nomination (or approval) mechanism, in which agents either approve or disapprove of a nomination, though that mechanism suffers from a possibility of returning fewer agents than required. In order to tackle potentially malicious (or just bad) reviewers, Lev et al. (2023) reworked PeerNomination so it weighs down suspected low-quality reviewers, while maintaining strategyproofness. On the other hand, Walsh (2014) suggested a PageRank-like peer-evaluation method, that gives up on strategyproofness in order to be able to weigh down less-regarded reviewers. In addition, several algorithms have been suggested involving how to deal with various biases and issues in agents' bidding and grading-normalization (Noothigattu, Shah, and Procaccia 2019; Stelmakh, Shah, and Singh 2019; Wang and Shah 2019).

To model the inaccuracies in reviewers' assessments, we assume that each agent is associated with a noisy observation of the ground truth according to a Mallows model (Mallows 1957). These have been widely used to compare peer-evaluation mechanisms (Aziz et al. 2016, 2019; Mattei, Turrini, and Zhydkov 2020; Lev et al. 2023).

Preliminaries

Our peer-evaluation agents will be denoted by $N = \{1, 2, \dots, n\}$. The agents in peer-evaluation represent both candidates that wish to be selected, as well as agents which vote by ranking candidates. Of the n candidates, we wish to select a subset of size k.

In this paper we will assume agents give grades to the candidates that they review[1]. Each agent i has a set $N_i \subseteq N$ of m agents that they review and grade, i.e., each agent has a function $A_i : N_i \to \mathbb{R}$, the grade it gives each paper.

As in previous peer-evaluation papers, we will assume there is a ground-truth, which is the ranking of the papers that we aim for. For simplicity, we assume the agents are numbered according to to their true ranking, so the top k agents are agents $\{1, 2, \dots, k\}$.

We use the Mallows model to reflect the noisy observation of the ground truth of each agent. The Mallows model is parameterized by a dispersion parameter $\phi \in [0, 1]$ and the ground truth ranking $\sigma \in \pi(N)$, for $\pi(N)$ being all possible orderings of N agents (as noted above, we assume that $\sigma = 1 \succ 2 \succ 3 \succ \dots \succ n-1 \succ n$). The Kendall-$\tau$ distance counts the number of pairwise disagreements between two rankings, and for any two rankings of the same N agents $a, b \in \pi(N)$, the value $d(a, b)$ shall denote the Kendall-τ distance between them. The Mallows model first builds a probability space from which agents' ranking are sampled.

[1] As noted above, other possibilities in the literature include approval/disapproval or rankings. Of course, numerical grades can be easily translated into a ranking.

For any ranking $r \in \pi(N)$, the probability of an agent having that ranking is $P(r) = P(r \mid \sigma, \phi) = \frac{1}{Z}\ \phi^{d(r,\sigma)}$, where $Z = 1 \cdot (1+\phi) \cdot (1+\phi+\phi^2) \dots (1+\phi+\dots+\phi^{n-1})$. For every agent $i \in N$, we select their own noisy ranking from this probability space[2].

Partition Algorithm

The Partition algorithm, as presented in Alon et al. (2011), takes the set of agents, divides them into two sets (in this paper we also ran simulations with 3 and 4 sets), and each agent reviews agents from a set that it is not a member of. We assume the existence of a function $Reviewer2Agents(C_1, C_2, m)$ that assigns agents from one set (C_1) to review m of the other set (C_2) and vice versa (any algorithm that does this can be used, as is assumed in Aziz et al. (2016, 2019); this can be done greedily, or, as detailed in Lev et al. (2023), based on Euler cycles). Note that while we assume (as is common in papers analyzing Partition) that clusters are created randomly, they can take into account any conflict of interest issue, by putting conflicted parties in the same cluster.

Algorithms

We describe two algorithms based on Partition: *Two-Stage Partition* (Algorithm 1) and *Two-Stage Flexible Partition* (Algorithm 2). Both algorithms try to improve the way we use agents' reviews by using the initial, first stage, information to decide which candidates are clearly good (or bad), and then focusing agents' reviews on the set of candidates in the "middle", i.e., for which our signal is not clear and obvious. The intuition behind this is that very good and very bad candidates can be detected with few reviews, while sifting between more borderline candidates is harder, and we prefer to have more reviews and information on those, to help us determine the correct ranking.

Therefore, instead of every agent reviewing m candidates and each candidate getting m reviews, agents still review m candidates, but not all candidates get m reviews. For example, in the first stage, all agents may review $\frac{m}{4}$ candidates, and each candidate gets $\frac{m}{4}$ reviews. Now, a few top candidates can be chosen without further reviews needed, and several bottom candidates can be rejected. The remaining $\frac{3}{4}m$ reviews each agent does are divided between the remaining candidates, giving them, overall, more than m reviews each.

The difference between the two algorithms is in how they treat the results of the first round. In Two-Stage Partition, the first round outcome – both the bottom rejected candidates and the top accepted candidates – are fixed. That is, they are either rejected or accepted, and the second stage involves the remaining candidates. In Two-Stage Flexible Partition, these groups can change. Following the first stage, we have top/bottom candidates groups according to their average reviews, and when each agent does an additional review, the candidates in these groups do not get an additional review. Following the additional single review from each agent, the average grade of candidates that were reviewed changes, and they may now be ranked in the top/bottom groups, while some candidates in those groups may no longer be so highly/poorly ranked, and therefore are no longer in the groups. Now, when each agent does an additional review, again, those in these groups do not get a review, but agents who "left" those groups will get one. That is, if paper x got very bad reviews and was in the bottom $f\%$ of papers, thus getting rejected, it may be that following additional reviews, paper y got sufficiently bad reviews to be ranked even lower than x. Thus, x got "pushed up", and now it will get another review. This process continues until all agents have given m reviews. The Two-Stage Flexible Partition is, as its name suggests, a more flexible mechanism, able to incorporate further data, and allows for updating of the first stage outcome so that initial perception may be corrected[3].

```
Algorithm 1: Two-Stage Two-Cluster Partition
Input: Set of agents $N$;
    $k$ number of agents to be chosen (we assume $k$ divisible by 2);
    $m$ number of reviews per agent;
    $f$ number of reviews in first round;
    $h$ size of the higher candidates group (chosen after first round);
    $l$ size of the lower candidates group (rejected after first round);
    $Reviewer2Agents(C_1, C_2, H, L, AgentReviewed, m')$ a
    function assigning each agent $i$ a group $N_i \subseteq N$, $|N_i| = m'$,
    that is from the cluster $i$ is not part of, such that $N_i \cap H = \emptyset$
    ($H$ is the top ranked set), $N_i \cap L = \emptyset$ ($L$ is the bottom ranked
    set), and $N_i \cap AgentReviewed[i] = \emptyset$ ($AgentReviewed[i]$
    is the set of candidates agent $i$ already reviewed).

Output: Set of accepted agents $W$.
Generate a partition $C_1$, $C_2$ of $N$ randomly
$s \leftarrow 2$                                   ▷ Number of rounds
$L \leftarrow \emptyset$;                          ▷ Current bottom candidates
$W \leftarrow \emptyset$;                          ▷ Top candidates
$m' \leftarrow f$   ▷ Number of reviews per candidate in current round
AgentReviewed = $(\emptyset, \dots, \emptyset)$
for $round = 0$ to $s$ do          ▷ Only two rounds in this algorithm
    $Reviewer2Agents(C_1, C_2, W, L, AgentReviewed, m')$
    for agent $i \in N$ do
        Agent reviews candidates $N_i$
        AgentReviewed[i] ← AgentReviewed[i] $\cup N_i$
    end for
    $W \leftarrow W \cup \{\{\frac{h}{2}$ top agents $\in C_1\} \cup \{\frac{h}{2}$ top agents $\in C_2\}\}$
    $L \leftarrow \{\frac{l}{2}$ bottom agents $\in C_1\} \cup \{\frac{l}{2}$ bottom agents $\in C_2\}$
    $C_1 \leftarrow C_1 \setminus (L \cup W)$
    $C_2 \leftarrow C_2 \setminus (L \cup W)$
    $m' \leftarrow m - f$
end for
return W
```

In the results section we will describe the efficiency of the new algorithms on various peer review task (different ϕ,

[2] To convert from each agent's ranking to its grade, we gave the top candidate 100, and the grades decrease, point by point, as a candidate is located lower in the ranking. The function A_i returns the grade, based on this ranking. This Borda-like score has been used in other peer-evaluation papers (Aziz et al. 2016, 2019)

[3] Note that while such a mechanism can be done when agents report a ranking of the candidates they have reviewed, or a grade, when using approval voting such a mechanism is not workable, as the number of approvals each candidate receives can only grow, making the flexible mechanism unusable.

Algorithm 2: Two-Stage Flexible Two-Cluster Partition

```
Input: Set of agents $N$; $k$ number of agents to be chosen (we
    assume $k$ divisible by 2);
    $m$ number of reviews per agent;
    $f$ number of reviews in first round;
    $h$ size of the higher candidates group (chosen after first round);
    $l$ size of the lower candidates group (rejected after first round);
    $Reviewer2Agents(C_1, C_2, H, L, AgentReviewed, m')$ a
    function assigning each agent $i$ a group $N_i \subseteq N$, $|N_i| = m'$,
    that is from the cluster $i$ is not part of, such that $N_i \cap H = \emptyset$
    ($H$ is the top ranked set), $N_i \cap L = \emptyset$ ($L$ is the bottom ranked
    set), and $N_i \cap AgentReviewed[i] = \emptyset$ ($AgentReviewed[i]$
    is the set of candidates agent $i$ already reviewed).
Output: Set of accepted agents $W$.
Generate a partition $C_1, C_2$ of $N$ randomly
$s \leftarrow m - f + 1$                          ▷ Number of rounds
$H \leftarrow \emptyset$;                         ▷ Current top candidates
$L \leftarrow \emptyset$;                         ▷ Current bottom candidates
$m' \leftarrow f$   ▷ Number of reviews per candidate in current round
AgentReviewed = $(\emptyset, \dots, \emptyset)$
for $round = 0$ to $s$ do
    $Reviewer2Agents(C_1, C_2, H, L, AgentReviewed, m')$
    for agent $i \in N$ do
        Agent reviews candidates $N_i$
        AgentReviewed[i] ← AgentReviewed[i] $\cup N_i$
    end for
    $H \leftarrow \{\frac{h}{2}$ top agents $\in C_1\} \cup \{\frac{h}{2}$ top agents $\in C_2\}$
    $L \leftarrow \{\frac{l}{2}$ bottom agents $\in C_1\} \cup \{\frac{l}{2}$ bottom agents $\in C_2\}$
    $m' \leftarrow 1$        ▷ After the first round, with $f$ reviews, all
        subsequent rounds have 1 review.
end for
$W \leftarrow \{\frac{k}{2}$ top agents $\in C_1\} \cup \{\frac{k}{2}$ top agents $\in C_2\}$
return W
```

m (number of reviews per candidate), and f, the number of chosen candidates following the first stage).

Observation 1. *Two-Stage Partition and Two-Stage Flexible Partition are strategyproof.*

Proof. All parameters are independent for each cluster. So the number of agents being accepted/rejected after each stage in each cluster is fixed, and not dependent on the outcomes in the other cluster. This means that an agent's reviews cannot affect its chance of acceptance: It does not influence the grade of any agent in its cluster, nor how many will be accepted or rejected at any stage. □

Empirical Setup

We built a simulator to compare Two-Stage (Flexible) Partition to the single-step, classic, Partition mechanism. We implemented all algorithms and compared their performance.

We measure the accuracy of an exact peer-selection mechanism by counting how many agents from the top k positions in the ground truth have been selected, as a proportion of all k agents selected. We will use True Positive Rate (TPR) (a.k.a *Recall*) to calculate each algorithm performance.

$$TPR := \frac{TP}{TP + FN}$$

For TP: all candidates that been chosen and are in the top k (according to ground truth); and FN: all candidates that been chosen but are not in the top k (according to ground truth).

Setup

Following in the footsteps of previous peer-selection comparisons (Aziz et al. 2019; Lev et al. 2023), we set n, the number of agents to 120. And we tested the algorithm on various values of:

ϕ Mallows dispersion parameter. Tested values: 0.75, 0.8, 0.85, 0.9, 0.95.

k Number of chosen agents. Tested values: 6, 12, 18, 24.

m the number of reviews per agent, (Tested values: 3, 5, 7, 9, 11, 15)

f the number of reviews in first round, (Tested values: $\frac{2m}{10}, \frac{3m}{10}, \frac{4m}{10}$)

h size of the higher candidates group (i.e., is chosen after the first round), (Tested values: $0, \frac{k}{10}, \frac{2k}{10}, \frac{3k}{10}, \frac{4k}{10}, \frac{k}{2}$)

l size of the lower candidates group (i.e., is eliminated after the first round), (Tested values: 60, 66, 72, 78, 84, 90)

We compare our both algorithms with 3 other algorithms:

Partition Regular, single-stage, Partition, with two clusters ($\ell = 2$), and each agent reviews the same number of agents as the two-stage mechanism.

Optimal Partition Single-stage Partition, with two clusters ($\ell = 2$), and each agent reviews all agents in the other cluster, so $m = \frac{n}{2}$.

Best Partition We partition the candidates into two clusters ($\ell = 2$) and select from each cluster the top $\frac{k}{2}$ according to the ground-truth ranking.

We repeated these experiments using Partition with more clusters ($\ell = 3$ and $\ell = 4$), which are a natural extension of the original Alon et al. (2011) algorithm, but as the results were similar to the two-cluster case ($\ell = 2$), we do not report them separately.

The experiment was repeated $10,000$ times for each setting. We compared all algorithms in each setting with the same clusters to be able to compare directly the effects of the algorithm and not the random assignment into clusters. We ran our Python code on a server with 2 Intel Xeon 6252 chips, each with 24 cores, running at 2.1 GHz; 1TB of RAM; running Oracle Linux Server.

Results

Our results can be divided into two parts: how the two-stage mechanism outcome compares to the single-stage one; and of the various parameters of the two-stage mechanism, which make the algorithm perform best.

Quality of Outcome

We see that both suggested algorithms achieved better results then the classic Partition for the same number of candidates reviewed by each agent. Figure 1 compares the candidates selected by each of the algorithms, showing that the new algorithms have a better chance of selecting middle

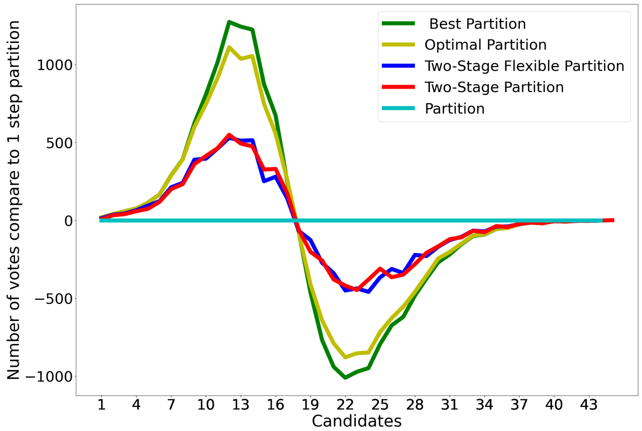
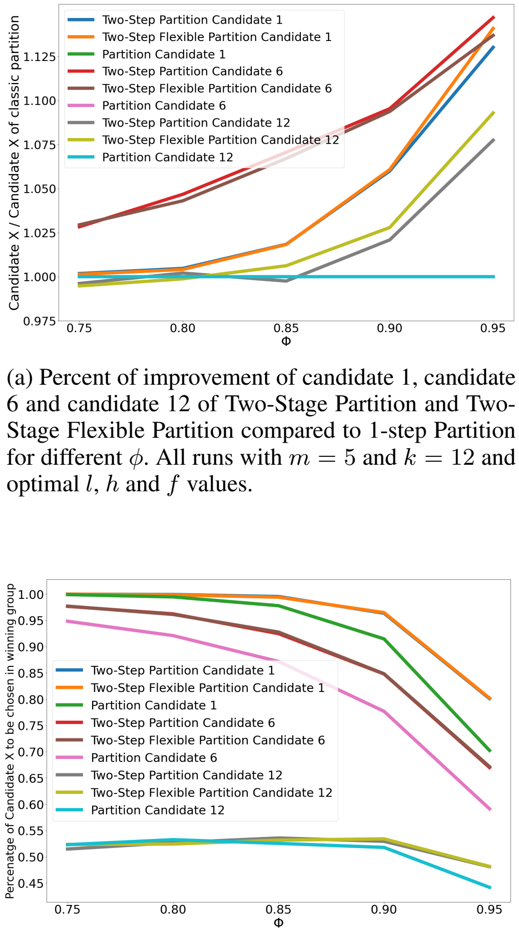


Figure 1: Number of times each candidate has been chosen in the winning set with Best Partition, Optimal Partition, Two-Stage Flexible Partition and Two-Stage Partition compared to classic Partition as the baseline. All runs with $\phi = 0.85$, $m = 5$ and $k = 18$.



(a) Percent of improvement of candidate 1, candidate 6 and candidate 12 of Two-Stage Partition and Two-Stage Flexible Partition compared to 1-step Partition for different ϕ. All runs with $m = 5$ and $k = 12$ and optimal l, h and f values.

(b) Percent of runs in which candidate 1, candidate 6 and candidate 12 where chosen in the winner group of Two-Stage Partition, Two-Stage Flexible Partition and Partition for different ϕ.

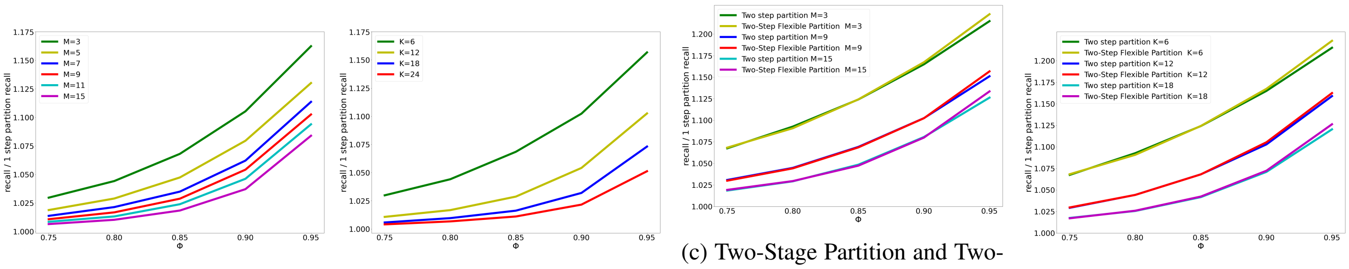
Figure 2: Candidates 1, 6, and 12 in Two-Stage Partition mechanisms vs. 1-Stage Partition. All runs with $m = 5$ and $k = 12$ and optimal l, h and f values.

ranking candidates from within the top-k compared to classic Partition. That is, the advantage of the new algorithms is not in accepting the top agent more often (it is almost always selected by classic Partition), nor better at selecting the kth agent or around that (selecting more often the $(k-1)$th agent or selecting less often the $(k+1)$th agent), but rather a higher chance of selecting the $\frac{k}{2}$th agent and agents in that general ranking. In Figure 3a we show that the improvement in the recall on candidates increases the noisier the information is (i.e., as ϕ grows), and it is more significant the smaller the number of reviews submitted by each agent (m) is. Examining significance, using χ^2 test we found that when $k = 6$, for $\phi \geq 0.7$ and $m \leq 11$, the improvement of the Two-Stage mechanisms is significant. For $k = 12$, for $\phi \geq 0.75$ and $m \leq 11$, the improvement over 1-step Partition is, again, significant. For $k = 18$, for $\phi \geq 0.8$ and $m \leq 11$, the improvement is significant. This does make intuitive sense, as the larger k grows, it is easier, even with mildly correlated rankings, to approximate the ground-truth. Hence, when k grows, the significant results happen with a larger ϕ, and when m is large enough, even Partition can get very good results, hence the lack of significance for those values. Examining specific candidates, we see our intuition regarding Figure 1 is correct – for example, for $k = 18, m = 5, \phi = 0.85$, the improvement of Two-Stage Partition over regular Partition is highly significant ($p < 0.001$) for candidates 3-17, very significant for candidate 2 ($p < 0.005$), and still significant for candidate 1 ($p < 0.05$).

Figures 3a and 3b compare the performance of the new algorithms depending on the size of the winner group (k), and on how many reviews are submitted by each agent (m). Of course, having all possible information is best, hence Partition with $\frac{n}{2}$ reviews dominates everything (but is not realistically implementable). We see that both new algorithms have a better recall than classical Partition, but when the k or m are small, the new algorithms have the most advantage over it. We also see a large advantage to the new algorithms when candidates' votes are more noisy, i.e., larger ϕ.

Figure 2a strengthens the statement above regarding the significant contribution of the two-stage mechanisms particularly to the mid-k candidates. We see that the middle candidate had the most improvement, and this advantage grew as the votes were less correlated (ϕ grew). Moreover, the raw numbers in Figure 2b show that this improvement over Partition on noisy votes, i.e., higher ϕ, is not due to a better performance on the noisy votes (unsurprisingly, it is easier to find the top k when votes are close to ground-truth), but rather due to a larger improvement over Partition, which becomes worse much faster in noisy settings.

When comparing the two new suggested algorithms, we find that the algorithms' fine-tuning parameters (e.g., size of f, etc.) are quite similar for both. But we see that when votes are less noisy, i.e, smaller ϕ, there is a slight advantage



(a) Two-Stage Partition compared to 1-step Partition for different m and ϕ values. All runs with $k = 12$.

(b) Two-Stage Partition compared to 1-step Partition for different k and ϕ values. All runs with $m = 9$.

(c) Two-Stage Partition and Two-Stage Flexible Partition compared to 1-step Partition for different m and ϕ values. All runs with $k = 18$.

(d) Two-Stage Partition compared to 1-step Partition for different k and ϕ values. All runs with $m = 3$.

Figure 3: Percent of improvement in recall compared to 1-step Partition. All runs with optimal f for that m, k, and ϕ.

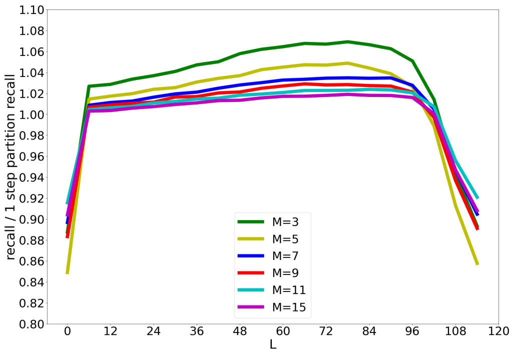


Figure 4: Percent of improvement of Two-Stage Partition recall over 1-step partition, for different values of l. All runs with $\phi = 0.85$, $k = 12$, $f = \frac{2m}{10}$, $h = 0$.

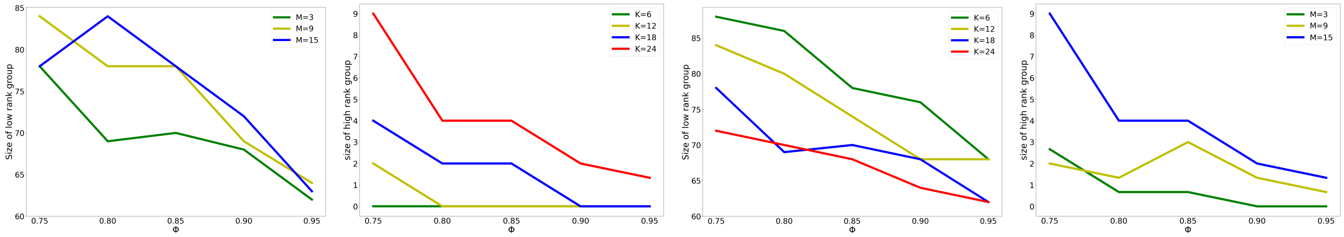
to Two-Stage Partition, and when votes are more noisy, i,e., larger ϕ, Two-Stage Flexible Partition has a slight advantage. This seems to imply that the flexibility of Two-Stage Flexible Partition does indeed come to play with noisy rankings, and is able to correct for a mediocre first round. Figures 3c and 3d compare between the improvement of the new algorithms over regular, 1-stage, Partition depending on the size of the winner group (k) and on the number of reviews (m). We see that as k grows and m is smaller, Two-Stage Flexible Partition achieves better results compared to Two-Stage Partition. This might be explained, as before, as higher k and lower m are more sensitive to errors with uncorrelated preferences (larger ϕ), making the flexibility meaningful.

Algorithm Fine-Tuning

We now examine the various parameters and see which values lead the Two-Stage Partition mechanisms to their best performance (as noted above, both algorithms are similar with regard to their optimal parameter choice). To do so, we wish to find when Two-Stage Partition recall was closest to regular Partition with everyone reviewing everyone else (Optimal Partition). Note that we are not comparing to the ground truth, as we wish to compare to the best information that is possible to glean from all possible reviews.

We assume ϕ, k and m are given as part of the problem description (e.g., peer evaluation of students in an online course may have higher ϕ than scientific conference reviews). In contrast to those, f, h and l are parameters that we can control and we can choose the best values for them given a task.

It seems that f (the number of reviews given in the first round) should to be pretty low ($\lceil \frac{2m}{10} \rceil$), as leads to better results in all our settings (see Figure 7). For h and l we see values change according to the scenario. For l (the size of the group of candidates we eliminate after the first stage), we see that is better to drop a large number of candidates after first round, ideally will be in range of $\frac{n}{2}$ to $\frac{3}{4}n$ candidates, with the precise value depending on the precise values of ϕ, k and m (see Figure 4). When k is smaller, a higher l seems to work better (as can seen in Figure 5c), perhaps because when we choose fewer candidates there is less likelihood they will end up in the bottom $\frac{3}{4}n$ after the first round, and as number of winners increase we need to be more cautious about those we eliminate. We found that higher ϕ values will lead to lower l values, probably since when reviewers are more noisy we also need be more cautious about the amount of data we use to eliminate agents. When m is large, a larger l works better (as can seen in Figure 5a), probably because the significant number of reviews means we have enough information about the candidate even from the first stage, allowing us to eliminate with confidence. For h (the size of the group of candidates we pass after the first stage), we see quite the opposite picture of that of l. It is better for h to be small, in range of $0 - \frac{k}{3}$ candidates, with the precise value depending on the precise ϕ, k and m (see Figure 6). When k is larger, a higher h seems to work better (as can seen in Figure 5b), perhaps because when we choose more candidates we probably will be in the top $\frac{k}{3}$ on the first round, and as number of winners decrease we need to be more cautious about those we choose. We found that higher ϕ values will lead to lower h values, probably since when reviewers are more noisy we also need be more cautious about the candidates we choose. When m is large, a larger h works better (as can seen in Figure 5d), probably because the significant number of reviews means we have enough information about the candidate even from the first stage, allowing us to choose with confidence.



(a) Best performing size of l (bottom-ranked set size) for different m and ϕ values. All runs with $k = 18$.

(b) Best performing size of h (top-ranked set size) for different k and ϕ values. All runs with $m = 15$.

(c) Best performing size of l (bottom-ranked set size) for different k and ϕ values. All runs with $m = 3$.

(d) Best performing size of h (top-ranked set size) for different m and ϕ values. All runs with $k = 24$.

Figure 5: Best performing size of top/bottom set for different values. All runs with optimal f for that m, k, and ϕ.

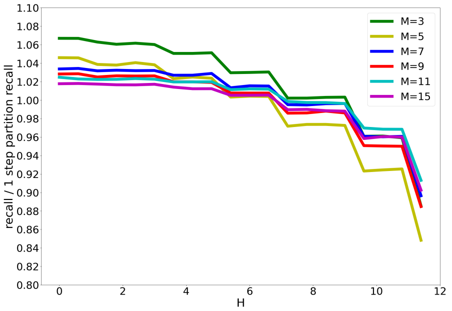


Figure 6: Percent of improvement of Two-Stage Partition recall over 1-step partition, for different values of h. All runs with $\phi = 0.85$, $k = 12$, $l = 0.7$, $f = 0.2$.

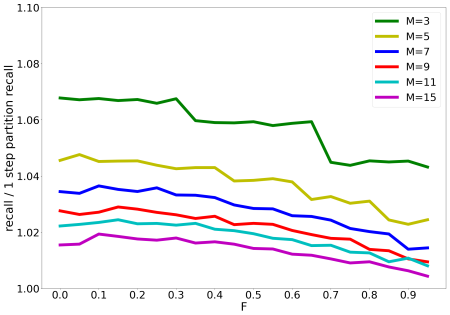


Figure 7: Percent of improvement of Two-Stage Partition recall over 1-step partition, for different values of f. All runs with $\phi = 0.85$, $k = 12$, $l = 0.7$, $h = 0$.

Discussion

In this paper we investigate using a two-stage mechanism for peer-evaluation. While the use of such mechanisms in the real-world has expanded in the past few years (Leyton-Brown et al. 2022), beyond the basic intuition behind it (focusing reviews on more “divisive” papers), there has not been, to our knowledge, any further investigation of this idea. Here, we took the most widely explored strategyproof mechanism – Partition – and examined its performance when adding a Two-Stage component to it, using two different methods to implement how the mechanisms decide on which candidates to focus (a fixed set vs. a flexible, changing set of papers).

While it seems the intuition is indeed correct, and focusing on a subset of papers does improve the performance of the peer-evaluation mechanism, the improvement was not where we expected it to be. We expected the “borderline” papers to be more exact. That is, that the paper ranked at $k-1$ will more surely be included vs the paper ranked at $k+1$. However, our simulations showed that this is not the key benefit of the Two-Stage mechanisms, but rather the more “middle-of-the-road” papers. Those ranked around $\frac{k}{2}$ benefited most, as their chance of being included in the winning set increased dramatically. It seems that borderline papers are hard to differentiate, even when getting more reviews; while the better papers were able to more clearly establish their quality.

In addition we were able to explore what parameters improve the algorithms’ performance best, depending on the values of m, k, and ϕ. Somewhat surprisingly, a fairly small benefit first stage suffices to help the algorithms’ performance, as long as enough papers are rejected. While this may seem counter-intuitive, it seems the limited signal in the first stage is enough such that the chances of getting enough reviews to counter it is of low-enough probability to not be worth it.

There are some obvious extensions to this work: first and foremost, examining if we see similar outcomes in other peer-evaluation mechanisms. We hypothesize that we will see something similar (e.g., the two stages help the middle-of-the-road papers the most), but this has yet to be examined. Furthermore, for other mechanisms a two-stage mechanism may not be as straightforwardly strategyproof, and may require a far more complex re-working of the algorithms to accommodate a two-stage system. Beyond this, examining outcomes in distribution that are not Mallows may lead to deeper understanding of the two-stage systems (though, so far, peer-evaluation papers, requiring a ground-truth to compare themselves to, focus on Mallows distribution for comparison and quality estimates).

References

Alon, N.; Fischer, F.; Procaccia, A. D.; and Tennenholtz, M. 2011. Sum of us: strategyproof selection from the selectors. In *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, 101–110. Groningen, The Netherlands.

Aziz, H.; Lev, O.; Mattei, N.; Rosenschein, J. S.; and Walsh, T. 2016. Strategyproof Peer Selection: Mechanisms, Analyses, and Experiments. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI)*, 397–403. Phoenix, Arizona.

Aziz, H.; Lev, O.; Mattei, N.; Rosenschein, J. S.; and Walsh, T. 2019. Strategyproof peer selection using randomization, partitioning, and apportionment. *Artificial Intelligence (AIJ)*, 275: 295–309.

Bjelde, A.; Fischer, F.; and Klimm, M. 2017. Impartial Selection and the Power of Up to Two Choices. *ACM Transactions on Economics and Computation (TEAC)*, 5(4): 1–20.

Bousquet, N.; Norin, S.; and Vetta, A. 2014. A Near-Optimal Mechanism for Impartial Selection. In *Proceedings of the 10th International Conference on Web and Internet Economics (WINE)*, 133–146. Beijing, China.

Fischer, F.; and Klimm, M. 2015. Optimal Impartial Selection. *SIAM Journal on Computing*, 44(5): 1263–1285.

Gao, X. A.; Wright, J. R.; and Leyton-Brown, K. 2019. Incentivizing evaluation with peer prediction and limited access to ground truth. *Artificial Intelligence (AIJ)*, 275: 618–638.

Holzman, R.; and Moulin, H. 2013. Impartial Nominations for a Prize. *Econometrica*, 81(1): 173–196.

Kurokawa, D.; Lev, O.; Morgenstern, J.; and Procaccia, A. D. 2015. Impartial Peer Review. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 582–588. Buenos Aires, Argentina.

Lawrence, N. D. 2022. The NeurIPS Experiment.

Lev, O.; Mattei, N.; Turrini, P.; and Zhydkov, S. 2023. PeerNomination: A novel peer selection algorithm to handle strategic and noisy assessments. *Artificial Intelligence (AIJ)*, 316: 103843.

Leyton-Brown, K.; Mausam; Nandwani, Y.; Zarkoob, H.; Cameron, C.; Newman, N.; and Raghu, D. 2022. Matching Papers and Reviewers at Large Conferences. ArXiV.

Mallows, C. L. 1957. Non-null ranking models. I. *Biometrika*, 44(1-2): 114–130.

Mattei, N.; Turrini, P.; and Zhydkov, S. 2020. PeerNomination: Relaxing Exactness for Increased Accuracy in Peer Selection. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 393–399. Yokohama, Japan.

Merrifield, M. R.; and Saari, D. G. 2009. Telescope time without tears: a distributed approach to peer review. *Astronomy & Geophysics*, 50(4): 4–16.

Noothigattu, R.; Shah, N. B.; and Procaccia, A. D. 2019. Choosing How to Choose Papers. Technical report, Carnegie Mellon University.

Shah, N. B. 2022. Challenges, experiments, and computational solutions in peer review. *Communications of the ACM*, 65(6): 76–87.

Stelmakh, I.; Shah, N. B.; and Singh, A. 2019. On Testing for Biases in Peer Review. Technical report, Carnegie Mellon University.

Walsh, T. 2014. The PeerRank Method for Peer Assessment. In *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI)*, 909–914. Prague, Czech Republic.

Wang, J.; and Shah, N. B. 2019. Your 2 is My 1, Your 3 is My 9: Handling Arbitrary Miscalibrations in Ratings. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 864–872. Montréal, Canada.

Xu, Y.; Zhao, H.; Shi, X.; and Shah, N. B. 2019. On Strategyproof Conference Peer Review. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 616–622. Macau.