Group Fairness in Peer Review[*]

Haris Aziz
UNSW Sydney
haris.aziz@unsw.edu.au

Evi Micha
University of Toronto
emicha@cs.toronto.edu

Nisarg Shah
University of Toronto
nisarg@cs.toronto.edu

Abstract

Large conferences such as NeurIPS and AAAI serve as crossroads of various AI fields, since they attract submissions from a vast number of communities. However, in some cases, this has resulted in a poor reviewing experience for some communities, whose submissions get assigned to less qualified reviewers outside of their communities. An often-advocated solution is to break up any such large conference into smaller conferences, but this can lead to isolation of communities and harm interdisciplinary research. We tackle this challenge by introducing a notion of group fairness, called the core, which requires that every possible community (subset of researchers) to be treated in a way that prevents them from unilaterally benefiting by withdrawing from a large conference.

We study a simple peer review model, prove that it always admits a reviewing assignment in the core, and design an efficient algorithm to find one such assignment. We use real data from CVPR and ICLR conferences to compare our algorithm to existing reviewing assignment algorithms on a number of metrics.

1 Introduction

Due to their large scale, conferences like NeurIPS and AAAI use an automated procedure to assign submitted papers to reviewers. Popular such systems include the Toronto Paper Matching System [1], Microsoft CMT[2], and OpenReview[3]. The authors submitting their works are often very interested in receiving meaningful and helpful feedback from their peers [2–4]. Thus, their overall experience with the conference heavily depends on the quality of reviews that their submissions receive.

The typical procedure of assigning papers to reviewers is as follows. First, for each paper-reviewer pair, a similarity score is calculated based on various parameters such as the subject area of the paper and the reviewer, the bids placed by the reviewer, etc. [1, 5–8]. Then, an assignment is calculated through an optimization problem, where the usual objectives are to maximize either the utilitarian social welfare, which is the total similarity score of all matched paper-reviewer pairs, or the egalitarian social welfare, which is the least total score of reviewers assigned to any submission. Relevant constraints are imposed to ensure that each submission receives an appropriate number of reviews, reviewer workloads are respected, and any conflicts of interest are avoided.

Peng et al. [9] recently mentioned that a major problem with the prestigious mega conferences is that they constitute the main venues for several communities, and as a result, in some cases, people are asked to review submissions that are beyond their main areas of work. They claim that a reasonable solution is to move to a de-centralized publication process by creating more specialized conferences appropriate for different communities. While specialized conferences definitely have their advantages, the maintenance of large conferences that attract multiple communities is also crucial for the emergence of interdisciplinary ideas that can be reviewed by diverse subject experts. Therefore, it is important to ensure that no group has an incentive to break off due to a feeling of being mistreated by the reviewing procedure of a large conference. In this work, we ask whether it

[*] A preliminary version was published in the proceedings of NeurIPS 2023.

[2] https://cmt3.research.microsoft.com/

[3] https://github.com/openreview/openreview-matcher

is possible to modify the existing reviewing processes to resolve this issue by treating the various communities satisfactorily. We clarify that the goal is not to build roadblocks to the creation of specialized conferences, but rather to mitigate the harm imposed on communities in large conferences.

To answer this, we look toward the literature on algorithmic fairness. Specifically, we adapt the group fairness notion known as *the core* [10]; to the best of our knowledge, we are the first to introduce it to the peer review setting. For a reviewing assignment to be in the core, it must ensure that no community (subset of researchers) can "deviate" by setting up its own conference in which (a) no author reviews her own submission, (b) each submission from within the community is reviewed by just as many reviewers as before, but now from within the community, (c) each reviewer reviews no more papers than before, and (d) the submissions are assigned to better reviewers, making the community happier. Intuitively, this is a notion of group fairness because it ensure that the treatment provided to every group of participants meets their "entitlement" (which is defined by what the group can achieve on its own). It is also a notion of stability (often also known as *core stability*) because it provides no incentives for any community to break off and get isolated by setting up their own conference instead. Note that this definition provides fairness to *every possible* community, and not only to predefined groups, as is the case for fairness definitions such as demographic parity and equalized odds that are popular in the machine learning literature [11]. In particular, it ensures fair treatment to emerging interdisciplinary communities even before they become visible.

1.1 Our Contribution

We consider a simple peer review model in which each agent submits (as the sole author) a number of papers to a conference and also serves as a potential reviewer. A reviewing assignment is valid if each paper is reviewed by k_p reviewers, each reviewer reviews no more than k_a papers, and no agent reviews her own submissions. To ensure that a valid assignment always exists, we assume that the maximum number of papers that each agent is allowed to submit is at most $\lfloor k_a/k_p \rfloor$.

In Section 3, we present an efficient algorithm that always returns a valid assignment in the core under minor conditions on the preferences of the authors. Specifically, our algorithm takes as input only the preference ranking of each author over individual potential reviewers for each of her submissions. Then, it produces an assignment that we prove to be in the core for any manner in which the agent's submission-specific preferences over individual reviewers may be extended to preferences over a set of k_p reviewers assigned to each submission, aggregated over submissions, subject to two mild conditions being satisfied.

In Section 4, we conduct experiments with real data from CVPR and ICLR conferences, and evaluate the price that our algorithm must pay — in lost utilitarian and egalitarian welfare — in order to satisfy the core and prevent communities from having an incentive to deviate. We also observe that reviewer assignment methods currently used in practice generate such adverse incentives quite often.

1.2 Related Work

As we mentioned above, usually the first step of a review assignment procedure is to calculate a similarity score for each pair of submission and reviewer which aims to capture the expertise of the reviewer for this submission. The problem of identifying the similarity scores has been extensively studied in the literature [1, 5–8, 12]. In this work, we assume that the similarity scores are given as an input to our algorithm after they have been calculated from a procedure that is considered as a black box. Importantly, our algorithm does not need the exact values of the similarity scores, but it only requires a ranking of the reviewers for each paper, indicating their relative expertise for this paper.

Given, the similarities scores various methods have been proposed for finding a reviewing assignment. The most famous algorithm is the Toronto Paper Matching System [1] which is a very broadly applied method and focuses on maximizing the utilitarian welfare, i.e., the sum of the similarities across all assigned reviewers and all papers. This approach has been adopted by other popular conference management systems such as EasyChair[4] and HotCRP [5] [13]. While this approach optimizes the total welfare, it is possible to discriminate against some papers. Therefore, other methods have focused on finding reviewing assignments that are (also) fair across all papers.

[4] https://easychair.org

[5] https://hotcrp.com

O'Dell et al. [14] suggest a method where the goal is to maximize the total score that a paper gets, while Stelmakh et al. [13] generalized this method by maximizing the minimum paper score, then maximizing the next smallest paper score, etc. Hartvigsen et al. [15] ensure fairness by requiring that each paper is assigned at least one qualified reviewer. Kobren et al. [16] proposed two algorithms that maximize that total utilitarian under the constraint that each paper should receive a score that exceeds a particular threshold. Payan and Zick [17] used the idea of envy-freeness [18] from the fair division literature to ensure fairness over the submissions. Moreover, some other works have focused on being fair over the reviewers rather than the papers [19, 20]). The core property that we consider in this work can be viewed as a fairness requirement over groups of authors. The reader can find more details about the challenges of the peer review problem in the recent survey of Shah [21].

In our model, the review assignment problem is related to exchange problems with endowments [22], since authors can be viewed as being endowed by their own papers which they wish to exchange with other authors that also serve as reviewers. For the basic exchange problem of housing reallocation, Shapley and Scarf [22] showed that an algorithm called *Top-Trading-Cycle (TTC)* finds an allocation which is in the core. The first part of our algorithm uses a variation of TTC where the agents (authors) are incorporated with multiple items (submissions), and constraints related to how many items each agent can get and to how many agents one item should be assigned should be satisfied. In contrast to classical exchange problem with endowments, our model has a distinctive requirement that agents/authors need to give away *all* their items/papers as the papers need to be reviewed by the agent who gets the paper. As we further explain in Section 3, this difference is crucial and requires further action from our algorithm than simply executing this variation of TTC. Various variations of TTC have been considered in the literature, tailored for different variations of the basic problem, but to the best of our knowledge, none of them can be directly applied in our model. To give an example, Suzuki et al. [23] consider the case that there are multiple copies of the same object and there are some quotas that should be satisfied, but they assume that each agent gets just one object while here each paper is assigned to multiple distinct reviewers.

2 Model

For $q \in \mathbb{N}$, define $[q] \triangleq \{1, \ldots, q\}$. There is a set of agents $N = [n]$. Each agent i submits a set of papers $P_i = \{p_{i,1}, \ldots, p_{i,m_i}\}$ for review by her peers, where $m_i \in \mathbb{N}$, and is available to review the submissions of her peers. We refer to $p_{i,\ell}$ as the ℓ-th submission of agent i; when considering the special case of each agent i having a single submission, we will drop ℓ and simply write p_i. Let $P = \cup_{i \in N} P_i$ be the set of all submissions and $m = \sum_{i \in N} m_i$ be the total number of submissions.

Assignment. Our goal is to produce a *(reviewing) assignment* $R : N \times P \to \{0, 1\}$, where $R(i, j) = 1$ if agent $i \in N$ is assigned to review submission $j \in P$. With slight abuse of notation, let $R_i^a = \{j \in P : R(i, j) = 1\}$ be the set of submissions assigned to agent i and $R_j^p = \{i \in N : R(i, j) = 1\}$ be the set of agents assigned to review submission j. We want the assignment to be *valid*, i.e., satisfy the following constraints:

- Each agent must be assigned at most k_a submissions for review, i.e., $|R_i^a| \leqslant k_a, \forall i \in N$.
- Each submission must be assigned to k_p agents, i.e., $|R_j^p| = k_p, \forall j \in P$.
- No agent should review one of her own submissions, i.e., $R(i, p_{i,\ell}) = 0, \forall i \in N, \ell \in [m_i]$.

To ensure that a valid assignment always exists, we impose the constraint that $m_i \cdot k_p \leqslant k_a$ for each $i \in N$, which implies that $m \cdot k_p \leqslant n \cdot k_a$. Intuitively, this demands that each agent submitting papers be willing to provide as many reviews as the number of reviews assigned to the submissions of any single agent. For further discussion on this condition, see Section 5.

Note that given $N' \subseteq N$ and $P'_i \subseteq P_i$ for each $i \in N'$ with $P' = \cup_{i \in N'} P'_i$, the validity requirements above can also be extended to a restricted assignment $\widehat{R} : N' \times P' \to \{0, 1\}$. Hereinafter, we will assume validity unless specified otherwise or during the process of building an assignment.

Preferences. Each agent $i \in N$ has a preference ranking, denoted $\sigma_{i,\ell}$, over the agents in $N \setminus \{i\}$ for reviewing her ℓ-th submission $p_{i,\ell}$.[6] These preferences can be based on a mixture of many factors, such as how qualified the other agents are to review submission $p_{i,\ell}$, how likely they are to provide

[6] Our algorithms continue to work with weak orders; one can arbitrarily break ties to convert them into strict orders before feeding them to our algorithms.

a positive review for it, etc. Let $\sigma_{i,\ell}(i')$ be the position of agent $i' \in N \setminus \{i\}$ in the ranking. We say that agent i prefers agent i' to agent i'' as a reviewer for $p_{i,\ell}$ if $\sigma_{i,\ell}(i') < \sigma_{i,\ell}(i'')$. Again, in the special case where the agents have a single submission each, we drop ℓ and just write σ_i. Let $\vec{\sigma} = (\sigma_{1,1}, \ldots, \sigma_{1,m_1}, \ldots, \sigma_{n,1}, \ldots, \sigma_{n,m_n})$.

While our algorithm takes $\vec{\sigma}$ as input, to reason about its guarantees, we need to define agent preferences over assignments by extending $\vec{\sigma}$. In particular, an agent is assigned a set of reviewers for each of her submissions, so we need to define her preferences over sets of sets of reviewers. First, we extend to preferences over sets of reviewers for a given submission, and then aggregate preferences across different submissions. Instead of assuming a specific parametric extension (e.g., additive preferences), we allow all possible extensions that satisfy two mild constraints; the group fairness guarantee of our algorithm holds with respect to any such extension.

Extension to a set of reviewers for one submission: Let $S \succ_{i,\ell} S'$ (resp., $S \succcurlyeq_{i,\ell} S'$) denote that agent i strictly (resp., weakly) prefers the set of agents S to the set of agents S' for her ℓ-th submission $p_{i,\ell}$. We require only that these preferences satisfy the following mild axiom.

Definition 1 (Order Separability)**.** For every disjoint $S_1, S_2, S_3 \subseteq N$ with $|S_1| = |S_2| > 0$, if it holds that $\sigma_{i,\ell}(i') < \sigma_{i,\ell}(i'')$ for each $i' \in S_1$ and $i'' \in S_2$, then we must have $S_1 \cup S_3 \succ_{i,\ell} S_2 \cup S_3$.

An equivalent reformulation is that between any two sets of reviewers S and T with $|S| = |T|$, ignoring the common reviewers in $S \cap T$, if the agent strictly prefers every (even the worst) reviewer in $S \setminus T$ to every (even the best) reviewer in $T \setminus S$, then the agent must strictly prefer S to T.

Example 1. Consider the common example of additive preferences, where each agent i has a utility function $u_{i,\ell} : N \setminus \{i\} \to \mathbb{R}_{\geqslant 0}$ over individual reviewers for her ℓ-th submission, inducing her preference ranking $\sigma_{i,\ell}$. In practice, these utilities are sometimes called similarity scores. Her preferences over sets of reviewers are defined via the additive utility function $u_{i,\ell}(S) \triangleq \sum_{i' \in S} u_{i,\ell}(i')$. It is easy to check that for any disjoint S_1, S_2, S_3 with $|S_1| = |S_2| > 0$, $u_{i,\ell}(i') > u_{i,\ell}(i'')$ for all $i' \in S_1$ and $i'' \in S_2$ would indeed imply $u_{i,\ell}(S_1 \cup S_3) > u_{i,\ell}(S_2 \cup S_3)$. Additive preferences are just one example from a broad class of extensions satisfying order separability.

Extension to assignments. Let us now consider agent preferences over sets of sets of reviewers, or equivalently, over assignments. Let $R \succ_i \widehat{R}$ (resp., $R \succcurlyeq_i \widehat{R}$) denote that agent i strictly (resp., weakly) prefers assignment R to assignment $\widehat{R}$. Note that these preferences collate the submission-wise preferences $\succ_{i,\ell}$ across all submissions of the agent. We require only that the preference extension satisfies the following natural property.

Definition 2 (Consistency)**.** For any assignment R, restricted assignment $\widehat{R}$ over any $N' \subseteq N$ and $P' = \cup_{i \in N'} P'_i$ (where $P'_i \subseteq P_i$ for each $i \in N'$), and agent $i^* \in N'$, if it holds that $R^p_{p_{i^*,\ell}} \succcurlyeq_{i^*,\ell} \widehat{R}^p_{p_{i^*,\ell}}$ for each $p_{i^*,\ell} \in P'_i$, then we must have $R \succcurlyeq_i \widehat{R}$.

In words, if an agent weakly prefers R to $\widehat{R}$ for the set of reviewers assigned to each of her submissions individually, then she must prefer R to $\widehat{R}$ overall.

Example 2. Let us continue with the previous example of additive utility functions. The preferences of agent i can be extended additively to assignments using the utility function $u_i(R) = \sum_{p_{i,\ell} \in P} u_{i,\ell}(R^p_{p_{i,\ell}})$. It is again easy to check that if $u_{i,\ell}(R^p_{p_{i,\ell}}) \geqslant u_{i,\ell}(\widehat{R}^p_{p_{i,\ell}})$ for each $p_{i,\ell}$, then $u_i(R) \geqslant u_i(\widehat{R})$. Hence, additive preferences are again one example out of a broad class of preference extensions that satisfy consistency.

Core. Our goal is to find a group-fair assignment which treats every possible group of agents at least as well as they could be on their own, thus ensuring that no subset of agents has an incentive to deviate and set up their own separate conference. Formally:

Definition 3 (Core)**.** An assignment R is in the core if there is no $N' \subseteq N$, $P'_i \subseteq P_i$ for each $i \in N'$, and restricted assignment $\widehat{R}$ over N' and $P' = \cup_{i \in N'} P'_i$ such that $\widehat{R} \succ_i R$ for each $i \in N'$.

In words, if any subset of agents deviate with any subset of their submissions and implement any restricted reviewing assignment, at least one deviating agent would not be strictly better off, thus eliminating the incentive for such a deviation. We also remark that our algorithm takes only the preference rankings over individual reviewers $\vec{\sigma}$ as input and produces an assignment R that is

```
ALGORITHM 1: CoBRA
Input: $N, P, \vec{\sigma}, k_a, k_p$
Output: $R$
$R, L, U$ =PRA-TTC$(N, P, \vec{\sigma}, k_a, k_p)$;
if $|U| > 0$ then
    $R$ =Filling-Gaps$(N, P, \vec{\sigma}, k_a, k_p, R, L, U)$;
```

```
ALGORITHM 2: PRA-TTC
Input: $N, P, \vec{\sigma}, k_a, k_p$
Output: $R, L, U$
$R(i, j) \leftarrow 0, \forall i \in N$ and $\forall j \in P$;
Construct the preference graph $G_R$;
while $\exists$ cycle in $G_R$ do
    Eliminate the cycle;
    Update $\overline{P}_i$-s by removing any completely assigned paper;
    Update $G_R$;
$U \leftarrow \{i \in N : \overline{P}_i \neq \emptyset\}$ ;
$L \leftarrow$ the last $k_p - |U| + 1$ agents in $N \setminus U$ to have all their submissions completely assigned ;
```

guaranteed to be in the core according to *every preference extension* of $\vec{\sigma}$ satisfying order separability and consistency.

3 CoBRA: An Algorithm for Computing Core-Based Reviewer Assignment

In this section, we prove our main result: when agent preferences are order separable and consistent, an assignment in the core always exists and can be found in polynomial time.

Techniques and key challenges: The main algorithm CoBRA (Core-Based Reviewer Assignment), presented as Algorithm 1, uses two other algorithms, PRA-TTC and Filling-Gaps, presented as Algorithm 2 and Algorithm 3, respectively. We remark that PRA-TTC is an adaptation of the popular Top-Trading-Cycles (TTC) mechanism, which is known to produce an assignment in the core for the house reallocation problem (and its variants) [22]. The adaptation mainly incorporates the constraints related to how many papers each reviewer can review and how many reviewers should review each paper. While for $k_p = k_a = 1$, PRA-TTC is identical with the classic TTC that is used for the house reallocation problem, the main difference of this problem with the review assignment problem is that in the latter each agent should give away her item (i.e., her submission) and obtain the item of another agent. Therefore, by simply executing TTC in the review assignment problem, one can get into a deadlock before producing a valid assignment. For example, consider the case of three agents, each with one submission. Each submission must receive one review ($k_p = 1$) and each agent provides one review ($k_a = 1$). The TTC mechanism may start by assigning agents 1 and 2 to review each other's submission, but this cannot be extended into a valid assignment because there is no one left to review the submission of agent 3. This is where Filling-Gaps comes in; it makes careful edits to the partial assignment produced by the PRA-TTC, and the key difficulty is to prove that this produces a valid assignment while still satisfying the core.

3.1 Description of CoBRA

Before we describe CoBRA in detail, let us introduce some more notation. Let $m^* = \max_{i \in N} m_i$. For reasons that will become clear later, we want to ensure that $m_i = m^*$, for each $i \in N$. To achieve that, we add $m^* - m_i$ dummy submissions to agent i, and the rankings over reviewers with respect to these submissions are arbitrarily. An assignment is called *partial* if there are submissions that are reviewed by less than k_p agents. A submission that is reviewed by k_p agents under a partial assignment is called *completely assigned*. Otherwise, it is called *incompletely assigned*. We denote with $\overline{P}_i(\widehat{R})$ the set of submissions of i that are incompletely assigned under a partial assignment $\widehat{R}$. We omit $\widehat{R}$ from the notation when it is clear from context.

CoBRA calls PRA-TTC, and then if needed, it calls Filling-Gaps. Below, we describe the algorithms.

ALGORITHM 3: Filling-Gaps

```
Input: $N, P, \vec{\sigma}, k_a, k_p, R, L, U$
Output: $R$
Phase 1;
Construct the greedy graph $G_R$;
while $\exists$ cycle do
    Eliminate the cycle ;
    Update $\overline{P}_i$-s by removing any completely assigned paper;
    Update $U$ and $L$ by moving any agent $i$ from $U$ to $L$ if $\overline{P}_i = \emptyset$;
    Update $G_R$;
Phase 2;
Construct the topological order $\vec{\rho}$ of $G_R$;
for $t \in [|U|]$ do
    while $\overline{P}_{\rho(t)} \neq \emptyset$ do
        Pick arbitrary $p_{\rho(t),\ell} \in \overline{P}_{\rho(t)}$ ;
        Find completely assigned $p_{i',\ell'}$, with $R(\rho(t), p_{i',\ell'}) = 0$, for some $i' \in U \cup L \setminus \{\rho(t)\}$ ;
        Find $i'' \neq \rho(t)$ such that $R(i'', p_{\rho(t),\ell}) = 0$ and $R(i'', p_{i',\ell'}) = 1$ ;
        $R(i'', p_{\rho(t),\ell}) \leftarrow 1$; $R(i'', p_{i',\ell'}) \leftarrow 0$; $R(\rho(t), p_{i',\ell'}) \leftarrow 1$ ;
        Remove $p_{\rho(t),\ell}$ from $\overline{P}_{\rho(t)}$ if it is completely assigned;
```

PRA-TTC. In order to define PRA-TTC, we first need to introduce the notion of a *preference graph*. Suppose we have a partial assignment $\widehat{R}$. Each agent i with $\overline{P}_i \neq \emptyset$ picks one of her incompletely assigned submissions arbitrarily. Without loss of generality, we assume that she picks her ℓ^*-th submission. We define the directed preference graph $G_{\widehat{R}} = (N, E_{\widehat{R}})$ where each agent is a node and for each i with $\overline{P}_i \neq \emptyset$, $(i, i') \in E_{\widehat{R}}$ if and only if i' is ranked highest in σ_{i,ℓ^*} among the agents that don't review p_{i,ℓ^*} and review less than k_a submissions. Moreover, for each $i \in N$ with $\overline{P}_i = \emptyset$, we add an edge from i to i', where i' is an arbitrary agent with $\overline{P}_{i'} \neq \emptyset$. PRA-TTC starts with an empty assignment, constructs the preference graph and searches for a directed cycle. If such a cycle exists, the algorithm eliminates it as following: For each (i, i') that is included in the cycle, it assigns submission p_{i,ℓ^*} to i' (if i's submissions are already completely assigned, it does nothing) and removes p_{i,ℓ^*} from $\overline{P}_i$, if it is now completely assigned. Then, the algorithm updates the preference graph and continues to eliminate cycles in the same way. When there are no left cycles in the preference graph, the algorithm terminates and returns two sets, U and L. The first set contains all the agents that some of their submissions are incompletely assigned and the set L contains the last $k_p - |U| + 1$ agents whose all submissions became completely assigned.

Filling-Gaps. CoBRA calls Filling-Gaps, if the U that returned from PRA-TTC is non empty. Before we describe the Filling-Gaps, we also need to introduce the notion of a *greedy graph*. Suppose that we have a partial assignment $\widehat{R}$ which indicates a set U that contains all the agents whose at least one submission is incompletely assigned. We define the directed greedy graph $G_{\widehat{R}} = (U, E_{\widehat{R}})$ where $(i, i') \in E_{\widehat{R}}$ if $\widehat{R}(i', p_{i,\ell}) = 0$ for some $p_{i,\ell} \in \overline{P}_i$. In other words, while in the preference graph, agent i points only to her favourite potential reviewer with respect to one of her incomplete submissions, in the greedy graph agent i points to any agent in $U \setminus \{i\}$ that could review at least one of her submissions that is incompletely assigned. Filling-Gaps consists of two phases. In the first phase, starting from the partial assignment R that was created from PRA-TTC, it constructs the greedy graph, searches for cycles and eliminates a cycle by assigning $p_{i,\ell}$ to agent i' for each (i, i') in the cycle that exists due to $p_{i,\ell}$ (when an edge exists due to multiple submissions, the algorithm chooses one of them arbitrary). Then, it updates $\overline{P}_i$ by removing any $p_{i,\ell}$ that became completely assigned and also updates U by moving any i to L if $\overline{P}_i$ became empty. It continues by updating the greedy graph and eliminating cycles in the same way. When no more cycles exist in the greedy graph, the algorithm proceeds to the second phase, where in $|U|$ rounds ensures that the incomplete submissions of each agent become completely assigned.

	Rounds				
Reviewers	1	2	3	4	5
1	p_3	p_2	p_4		
2	p_1	p_3		p_6	
3	p_2	p_1			p_4
4			p_1		p_5
5				p_2	p_3
6				p_5	

(a) Execution of PRA-TTC

	Rounds		
Reviewers	1	1	2
1	p_3, p_2, p_4	~~p_3~~, p_6 p_2, p_4	p_6 p_2, ~~p_4~~, p_5
2	p_1, p_3, p_6	p_1, p_3, p_6	p_1, p_3, p_6
3	p_2, p_1, p_4	p_2, p_1, p_4	p_2, p_1, p_4
4	p_1, p_5, p_6	p_1, p_5, p_6	p_1, p_5, p_6
5	p_2, p_3	p_2, p_3	p_2, p_3, p_4
6	p_5, p_4	p_5, p_4, p_3	p_5, p_4, p_3
	Phase 1	Phase 2	

(b) Execution of Filling-Gaps

Figure 1: Execution of CoBRA when $n = 6$, $k_p = k_a = 3$, $\sigma_1 = 2 \succ 3 \succ 4 \succ \ldots$, $\sigma_2 = 3 \succ 1 \succ 5 \succ \ldots$, $\sigma_3 = 1 \succ 2 \succ 5 \succ \ldots$, $\sigma_4 = 1 \succ 3 \succ 5 \succ \ldots$, $\sigma_5 = 6 \succ 4 \succ \ldots$ and $\sigma_6 = 2 \succ \ldots$. On the left table, we see the assignments that are established in each round of PRA-TTC by eliminating cycles. After the execution of PRA-TTC, three papers, p_4, p_5, p_6 are not completely assigned. Thus, $U = \{4, 5, 6\}$ and $L = \{3\}$. On the right table, we see the execution of Filling-Gaps. There is a cycle in the greedy graph which is eliminated at the first round of Phase 1. In Phase 2, where $\vec{\rho} = (6, 5)$, at the first round, since p_3 is authored by an agent in $U \cup L \setminus \{6\}$, is not reviewed by 6 and is completely assigned, p_3 is assigned to 6 while it is removed form 1 in which p_6 is now assigned. At the second round, since p_4 is authored by an agent in $U \cup L \setminus \{5\}$, is not reviewed by 5 and is completely assigned, p_4 is assigned to 5 while it is removed form 1 in which p_5 is now assigned.

3.2 Execution of CoBRA on a Toy Example

Figure 1 shows an execution of CoBRA on a small instance. Let us briefly describe how we get this assignment.

In the preference graph, we see that 1 has an outgoing edge to agent 2, agent 2 has an outgoing edge to agent 3 and agent 3 has an outgoing edge to agent 1. By eliminating this cycle we assign p_3 to 1, p_2 to 3 and p_1 to 2. By continuing to detect cycles in the preference graph given the preferences, we conclude on the partial assignment shown in the table on the left. For the table on the right, at the first phase of Filling-Gaps, when we create the greedy graph, we see that it consists of 3 nodes, since $U = \{4, 5, 6\}$, 4 has outgoing edges to 5 and 6 (since none of them review p_4), 6 has outgoing edges to 4 and 5, and 5 has no outgoing edge. Hence there is only one cycle and that is why p_6 is assigned to 4 and p_4 is assigned to 6. Then, 4 is moved to L, since p_4 becomes completely assigned. Then, there is no cycle in the greedy graph as while 6 has an outgoing edge to 5, 5 has no outgoing edges. Hence, we proceed to the second phase where there are two non-completely assigned papers, p_5 and p_6. $\vec{\rho} = (5, 6)$ as agent 6 reviews p_5, but 5 does not review p_6. Then, the algorithm makes sure that p_6 and then p_5 become completely assigned as it is described in the caption of the figure.

3.3 Main Result

We are now ready to present our main result.

Theorem 1. *When agent preferences are order separable and consistent, CoBRA returns an assignment in the core in $O(n^3)$ time complexity.*

Proof. First, in the next lemma, we show that CoBRA returns a valid assignment. The proof is quite non-trivial, several pages long, and tedious, so we defer it to the appendix.

Lemma 1. *CoBRA returns a valid assignment.*

Now, we show that the final assignment R that CoBRA returns is in the core. Note that while it is possible that an assignment of a submission of an agent in $U \cup L$, that was established during the execution of PRA-TTC, to be removed in the execution of Filling-Gaps, this never happens for submissions that belong to some agent in $N \setminus (U \cup L)$. For the sake of contradiction, assume that $N' \subseteq N$, with $P'_i \subseteq P_i$ for each $i \in N'$, deviate to a restricted assignment $\widetilde{R}$ over N' and $\cup_{i \in N'} P'_i$.

Note that $\widetilde{R}$ is valid only if $|N'| > k_p$, as otherwise there is no way each submission in $\cup_{i \in N'} P'_i$ to be completely assigned, since no agent can review her own submissions.

We distinguish into two cases and we show that in both cases the assignment is in the core.

Case I: $\exists i \in N' : i \notin L \cup U$**.** Let $i^* \in N'$ be the first agent in N' whose all submissions became completely assigned in the execution of PRA-TTC. Note that since there exists $i \notin U \cup L$, we get that $i^* \notin U \cup L$ from the definitions of U and L. Now, consider any $p_{i^*,\ell}$. Let $Q_1 = R^p_{p_{i^*,\ell}} \setminus (R^p_{p_{i^*,\ell}} \cap \widetilde{R}^p_{p_{i^*,\ell}})$ and $Q_2 = \widetilde{R}^p_{p_{i^*,\ell}} \setminus (R^p_{p_{i^*,\ell}} \cap \widetilde{R}^p_{p_{i^*,\ell}})$. If $Q_1 = \emptyset$, then we have that $R^p_{p_{i^*,\ell}} = \widetilde{R}^p_{p_{i^*,\ell}}$ which means that $R^p_{p_{i^*,\ell}} \succsim_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$. Otherwise, let $i' = \operatorname{argmax}_{i \in Q_1} \sigma_{i^*,\ell}(i)$, i.e., i' is ranked at the lowest position in $\sigma_{i^*,\ell}$ among the agents that review $p_{i^*,\ell}$ under R but not under $\widetilde{R}$. Moreover, let $i'' = \operatorname{argmin}_{i \in Q_2} \sigma_{i^*,\ell}(i)$, i.e., i'' is ranked at the highest position in $\sigma_{i^*,\ell}$ among the agents that review $p_{i^*,\ell}$ under $\widetilde{R}$ but not under R. We have $R(i', p_{i^*,\ell}) = 1$, if and only if i^* has an outgoing edge to i' at some round of PRA-TTC. At the same round, we get that i'' can review more submissions, since $i'' \in N'$ and if i^* has incompletely assigned submissions, then any $i \in N'$ has incompletely assigned submissions, and hence $|R^a_{i''}| < k_p \cdot m^* \leqslant k_a$. This means that if $\sigma_{i^*,\ell}(i') > \sigma_{i^*,\ell}(i'')$, then i^* would point i'' instead of i'. We conclude that $\sigma_{i^*,\ell}(i') < \sigma_{i^*,\ell}(i'')$. Then, from the definition of i' and i'' and from the order separability property we have that $R^p_{p_{i^*,\ell}} \succ_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$. Thus, either if Q_1 is empty or not, we have that for any $p_{i^*,\ell} \in P'_i$, it holds that $R^p_{p_{i^*,\ell}} \succsim_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$ and from consistency, we get that $R \succsim_{i^*} \widetilde{R}$ which is a contradiction.

Case II: $\nexists i \in N' : i \notin L \cup U$**.** In this case we have that $N' = U \cup L$, as $|U \cup L| = k_p + 1$. This means that for each $i \in U \cup L$ and $\ell \in [m^*]$, $\widetilde{R}^p_{p_{i,\ell}} = (U \cup L) \setminus \{i\}$. Let $i^* \in L$ be the first agent in L whose all submissions became completely assigned in the execution of PRA-TTC. Consider any $p_{i^*,\ell}$. Note that it is probable that while $p_{i^*,\ell}$ was assigned to some agent i in PRA-TTC, it was moved to another agent i' during the execution of Filling-Gaps. But, i' belongs to U and we can conclude that if $p_{i^*,\ell}$ is assigned to some $i \in N \setminus U$ at the output of CoBRA, this assignment took place during the execution of PRA-TTC. Now, let $Q_1 = R^p_{p_{i^*,\ell}} \setminus (R^p_{p_{i^*,\ell}} \cap \widetilde{R}^p_{p_{i^*,\ell}})$ and $Q_2 = \widetilde{R}^p_{p_{i^*,\ell}} \setminus (R^p_{p_{i^*,\ell}} \cap \widetilde{R}^p_{p_{i^*,\ell}})$. If $Q_1 = \emptyset$, then we have that $R^p_{p_{i^*,\ell}} = \widetilde{R}^p_{p_{i^*,\ell}}$ which means that $R^p_{p_{i^*,\ell}} \succsim_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$. If $Q_1 \neq \emptyset$, then $Q_1 \subseteq N \setminus (U \cup L)$ and $Q_2 \subseteq U \cup L$ since $\widetilde{R}^p_{p_{i^*,\ell}} = U \cup L$. Let $i' = \operatorname{argmax}_{i \in Q_1} \sigma_{i^*,\ell}(i)$, i.e., i' is ranked at the lowest position in $\sigma_{i^*,\ell}$ among the agents that review $p_{i^*,\ell}$ under R but not under $\widetilde{R}$. Moreover, let $i'' = \operatorname{argmin}_{i \in Q_2} \sigma_{i^*,\ell}(i)$, i.e., i'' is ranked at the highest position in $\sigma_{i^*,\ell}$ among the agents that review $p_{i^*,\ell}$ under $\widetilde{R}$ but not under R. From above, we know that the assignment of $p_{i^*,\ell}$ to i' was implemented during the execution of PRA-TTC, since $i' \in N \setminus (U \cup L)$. Hence, with very similar arguments as in the previous case, we will conclude that $\sigma_{i^*,\ell}(i') < \sigma_{i^*,\ell}(i'')$. We have $R(i', p_{i^*,\ell}) = 1$ if and only if i^* has an outgoing edge to i' at some round of PRA-TTC. At this round, we know that i'' can review more submissions, since $i'' \in N'$ and if i^* has incompletely assigned submissions, then any $i \in N'$ has incompletely assigned submissions. This means that if $\sigma_{i^*,\ell}(i') > \sigma_{i^*,\ell}(i'')$, then i^* would point i'' instead of i'. Hence, we conclude that $\sigma_{i^*,\ell}(i') < \sigma_{i^*,\ell}(i'')$. Then, from the definition of i' and i'' and from the order separability property we have that $R^p_{p_{i^*,\ell}} \succ_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$. Thus, either if Q_1 is empty or not, we have that for any $p_{i^*,\ell} \in P'_i$, it holds that $R^p_{p_{i^*,\ell}} \succsim_{i^*,\ell} \widetilde{R}^p_{p_{i^*,\ell}}$ and from consistency we get that $R \succsim_{i^*} \widetilde{R}$ which is a contradiction.

Lastly, we analyze the time complexity of CoBRA. First, we consider the time complexity of PRA-TTC. In each iteration, the algorithm assigns at least one extra reviewer to at least one incompletely-assigned submission. This can continue for at most $m \cdot k_p \leqslant n \cdot k_a$ iterations, since each submission should be reviewed by k_p reviewers. In each iteration, it takes $O(n)$ time to find and eliminate a cycle in the preference graph. Then, it takes $O(n^2)$ time to update the preference graph, since for each arbitrarily-picked incompletely-assigned submission of each agent, we need to find the most qualified reviewer who can be additionally assigned to it. By all the above, we conclude that the runtime of PRA-TTC is $O(n^3)$, by ignoring k_a which is a small constant in practice. After PRA-TTC terminates, CoBRA calls the Filling-Gaps algorithm. However, Lemma 3 ensures that at the end of PRA-TTC, $|L \cup U| \leqslant k_p + 1$, which is also a small constant. And Filling-Gaps only makes local changes that affect these constantly many agents. As such, the running time of Filling-Gaps is constant as well. Therefore, the time complexity of CoBRA is $O(n^3)$. □

Dataset	Alg	USW	ESW	α-Core		CV-Pr
				#unb-α	α^*	
CVPR '17	CoBRA	1.225 ± 0.021	0.000 ± 0.000	0%	1.000 ± 0.000	0%
	TPMS	1.497 ± 0.019	0.000 ± 0.000	89%	3.134 ± 0.306	100%
	PR4A	1.416 ± 0.019	0.120 ± 0.032	51%	1.700 ± 0.078	100%
CVPR '18	CoBRA	0.224 ± 0.004	0.004 ± 0.001	0%	1.000 ± 0.000	0%
	TPMS	0.286 ± 0.005	0.043 ± 0.004	0%	1.271 ± 0.038	100%
	PR4A	0.282 ± 0.005	0.099 ± 0.001	0%	1.139 ± 0.011	100%
ICLR '18	CoBRA	0.166 ± 0.001	0.028 ± 0.001	0%	1.000 ± 0.000	0%
	TPMS	0.184 ± 0.001	0.048 ± 0.002	0%	1.048 ± 0.008	90%
	PR4A	0.179 ± 0.001	0.082 ± 0.001	0%	1.087 ± 0.009	100%

Table 1: Results on CVPR 2017 and 2018, and ICLR 2018.

4 Experiments

In this section, we empirically compare CoBRA to TPMS [1], which is widely used (for example, it was used by NeurIPS for many years), and PR4A [13], which was used in ICML 2020 [24]. As mentioned in the introduction, these algorithms assume the existence of a similarity or affinity score for each pair of reviewer i and paper j, denoted by $S(i, j)$. The score (or utility) of a paper under an assignment R, denoted by u_j^p, is computed as $u_j^p \triangleq \sum_{i \in R_j^p} S(i, j)$. TMPS finds an assignment R that maximizes the utilitarian social welfare (USW), i.e., the total paper score $\sum_{j \in P} u_j^p$, whereas PR4A finds an assignment that maximizes the egalitarian social welfare (ESW), i.e., the minimum paper score $\min_{j \in P} u_j^p$.[7] We use $k_a = k_p = 3$ in these experiments.[8]

Datasets. We use three conference datasets: from the Conference on Computer Vision and Pattern Recognition (CVPR) in 2017 and 2018, which were both used by Kobren et al. [16], and from the International Conference on Learning Representations (ICLR) in 2018, which was used by Xu et al. [25]. In the ICLR 2018 dataset, similarity scores and conflicts of interest are also available. While a conflict between a reviewer and a paper does not necessarily indicate authorship, it is the best indication we have available, so, following Xu et al. [25], we use the conflict information to deduce authorship. Since in our model each submission has one author, and no author can submit more than $\lfloor k_a/k_p \rfloor = 1$ papers, we compute a maximum cardinality matching on the conflict matrix to find author-paper pairs, similarly to what Dhull et al. [26] did. In this way, we were able to match 883 out of the 911 papers. We disregard any reviewer who does not author any submissions, but note that the addition of more reviewers can only improve the results of our algorithm since these additional reviewers have no incentive to deviate. For the CVPR 2017 and CVPR 2018 datasets, similarity scores was available, but not the conflict information. In both these datasets, there are fewer reviewers than papers. Thus, we constructed artificial authorship relations by sequentially processing papers and matching each paper to the reviewer with the highest score for it, if this reviewer is still unmatched. In this way, we were able to match 1373 out of 2623 papers from CVPR 2017 and 2840 out of 5062 papers from CVPR 2018. In the ICLR 2018 and CVPR 2017 datasets, the similarity scores take values in $[0, 1]$, so we accordingly normalized the CVPR 2018 scores as well.

Measures. We are most interested in measuring the extent to which the existing algorithms provide incentives for communities of researchers to deviate. To quantify this, we need to specify the utilities of the authors. We assume that they are additive, i.e., the utility of each author in an assignment is the total similarity score of the $k_p = 3$ reviewers assigned to their submission.

Core violation factor: Following the literature [27], we measure the multiplicative violation of the core (if any) that is incurred by TPMS and PR4A (CoBRA provably does not incur any). This is done by computing the maximum value of $\alpha \geqslant 1$ for which there exists a subset of authors such that by deviating and implementing some valid reviewing assignment of their papers among themselves,

[7] Technically, subject to this, it maximizes the second minimum paper score, and then the third minimum paper score, etc. This refinement is also known as leximin in the literature.

[8] Note that $k_p = 3$ reviews per submission is quite common, although the reviewer load k_a is typically higher in many conferences, often closer to 6. However, the differences between different algorithms diminish with higher values of k_a.

they can each improve their utility by a factor of at least α. This can easily be formulated as a binary integer linear program (BILP). Because this optimization is computationally expensive (the most time-consuming component of our experiments), we subsample 100 papers[9] from each dataset in each run, and report results averaged over 100 runs. Note that whenever there exists a subset of authors with zero utility each in the current assignment who can deviate and receive a positive utility each, the core deviation α becomes infinite. We separately measure the percentage of runs in which this happens (in the column #unb-α), and report the average α among the remaining runs in the α^* column.

Core violation probability: We also report the percentage of runs in which a core violation exists (i.e., there exists at least one subset of authors who can all strictly improve by deviating from the current assignment). We refer to this as the *core violation probability* (CV-Pr).

Social welfare: Finally, we also measure the utilitarian and egalitarian social welfare (USW and ESW) defined above, which are the objectives maximized by TPMS and PR4A, respectively.

Results. The results are presented in Table 1. As expected, TPMS and PR4A achieve the highest USW and ESW, respectively, on all datasets because they are designed to optimize these objectives. In CVPR 2017, CoBRA and TPMS always end up with zero ESW because this dataset includes many zero similarity scores, but PR4A is able to achieve positive ESW. In all datasets, CoBRA achieves a relatively good approximation with respect to USW, but this is not always the case with respect to ESW. For example, in CVPR 2018, CoBRA achieves 0.004 ESW on average whereas PR4A achieves 0.099 ESW on average. This may be due to the fact that this dataset also contains many zero similarity scores, and the myopic process of CoBRA locks itself into a bad assignment, which the global optimization performed by PR4A avoids.

While CoBRA suffers some loss in welfare, TPMS and PR4A also generate significant adverse incentives. They incentivize at least one community to deviate in almost every run of each dataset (CV-Pr). While the magnitude of this violation is relatively small when it is finite (except for TPMS in CVPR 2017), TPMS and PR4A also suffer from unbounded core violations in more than half of the runs for CVPR 2017; this may again be due to the fact that many zero similarity scores lead to deviations by groups where each agent has zero utility under the assignments produced by these algorithms.

Of all these results, the high probability of core violation under TPMS and PR4A is perhaps the most shocking result; when communities regularly face adverse incentives, occasional deviations may happen, which can endanger the stability of the conference. That said, CoBRA resolves this issue at a significant loss in fairness (measured by ESW). This points to the need for finding a middle ground where adverse incentives can be minimized without significant loss in fairness or welfare.

5 Discussion

In this work, we propose a way for tackling the poor reviewing problem in large conferences by introducing the concept of "core" as a notion of group fairness in the peer review process. This fairness principle ensures that each subcommunity is treated at least as well as it would be if it was not part of the larger conference community.

We show that under certain —albeit sometimes unrealistic—assumptions , a peer review assignment in the core always exists and can be efficiently found. In the experimental part, we provide evidence that peer review assignment procedures that are currently used in practice, quite often motivate subcommunities to deviate and build their own conferences.

Our theoretical results serve merely as the first step toward using it to find reviewer assignments that treat communities fairly and prevent them from deviating. As such, our algorithm has significant limitations that must be countered before it is ready for deployment in practice. A key limitation is that it only works for single-author submissions, which may be somewhat more realistic for peer review of grant proposals, but unrealistic for computer science conferences. We also assume that each author serves as a potential reviewer; while many conferences require this nowadays, exceptions must be allowed in special circumstances. We also limit the number of submissions by any author to

[9] In Table 2 in the appendix, we report USW and ESW without any subsampling and we note that the qualitative relationships between the different algorithms according to each metric remain the same.

be at most $\lfloor k_a/k_p \rfloor$, which is a rather small value in practice, and some authors ought to submit more papers than this. We need to make this assumption to theoretically guarantee that a valid assignment exists. An interesting direction is to design an algorithm that can produce a valid assignment in the (approximate) core whenever it exists. Finally, deploying group fairness in real-world peer review processes may require designing algorithms that satisfy it approximately at minimal loss in welfare, as indicated by our experimental results.

References

[1] Laurent Charlin and Richard Zemel. The Toronto paper matching system: an automated paper-reviewer assignment system. In *Proceedings of the ICML Workshop on Peer Reviewing and Publishing Models*, 2013.

[2] G. David L. Travis and Harry M. Collins. New light on old boys: Cognitive and institutional particularism in the peer review system. *Science, Technology, & Human Values*, 16(3):322–341, 1991.

[3] Raymond S. Nickerson. What authors want from journal reviewers and editors. *American Psychological*, 60(6):661–662, 2005.

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

[5] David Mimno and Andrew McCallum. Expertise modeling for matching papers with reviewers. In *Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 500–509, 2007.

[6] Xiang Liu, Torsten Suel, and Nasir Memon. A robust model for paper reviewer assignment. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys)*, pages 25–32, 2014.

[7] Marko A. Rodriguez and Johan Bollen. An algorithm to determine peer-reviewers. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, pages 319–328, 2008.

[8] Hong Diep Tran, Guillaume Cabanac, and Gilles Hubert. Expert suggestion for conference program committees. In *Proceedings of the 11th International Conference on Research Challenges in Information Science (RCIS)*, pages 221–232, 2017.

[9] Andi Peng, Jessica Zosa Forde, Yonadav Shavit, and Jonathan Frankle. Strengthening subcommunities: Towards sustainable growth in AI research. arXiv:2204.08377, 2022.

[10] Donald Bruce Gillies. *Some theorems on n-person games*. Princeton University, 1953.

[11] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 3315–3323, 2016.

[12] Ivan Stelmakh, John Wieting, Graham Neubig, and Nihar B Shah. A gold standard dataset for the reviewer assignment problem. arXiv:2303.16750, 2023.

[13] Ivan Stelmakh, Nihar B. Shah, and Aarti Singh. PeerReview4All: Fair and accurate reviewer assignment in peer review. In *Proceedings of the 30th International Conference on Algorithmic Learning Theory (ALT)*, pages 828–856, 2019.

[14] Regina O'Dell, Mirjam Wattenhofer, and Roger Wattenhofer. The paper assignment problem. *Technical Report/ETH Zurich, Department of Computer Science*, 491, 2005.

[15] David Hartvigsen, Jerry C Wei, and Richard Czuchlewski. The conference paper-reviewer assignment problem. *Decision Sciences*, 30(3):865–876, 1999.

[16] Ari Kobren, Barna Saha, and Andrew McCallum. Paper matching with local fairness constraints. In *Proceedings of the 25th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1247–1257, 2019.

[17] Justin Payan and Yair Zick. I will have order! optimizing orders for fair reviewer assignment. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 440–446, 2022.

[18] Duncan Karl Foley. *Resource allocation and the public sector*. Yale University, 1966.

[19] Naveen Garg, Telikepalli Kavitha, Amit Kumar, Kurt Mehlhorn, and Julián . Mestre. Assigning papers to referees. *Algorithmica*, 58(1):119–136, 2010.

[20] Jing Wu Lian, Nicholas Mattei, Renee Noble, and Toby Walsh. The conference paper assignment problem: Using order weighted averages to assign indivisible goods. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1138–1145, 2018.

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

[22] Lloyd Shapley and Herbert Scarf. On cores and indivisibility. *Journal of Mathematical Economics*, 1(1):23–37, 1974.

[23] Takamasa Suzuki, Akihisa Tamura, and Makoto Yokoo. Efficient allocation mechanism with endowments and distributional constraints. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 50–58, 2018.

[24] Ivan Stelmakh. Towards fair, equitable, and efficient peer review. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, pages 15736–15737, 2021.

[25] Yichong Xu, Xiaofei Zhao, Han Shi, and Nihar B. Shah. On strategyproof conference peer review. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 616–622, 2019.

[26] Komal Dhull, Steven Jecmen, Pravesh Kothari, and Nihar B. Shah. Strategyproofing peer assessment via partitioning: The price in terms of evaluators' expertise. In *Proceedings of the 10th AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, pages 53–63, 2022.

[27] Brandon Fain, Kamesh Munagala, and Nisarg Shah. Fair allocation of indivisible public goods. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, pages 575–592, 2018.

Appendix

A Proof of Lemma 1

We want to prove that CoBRA returns a valid assignment. We start by showing that during the execution of PRA-TTC and the execution the first phase of Filling-Gaps, it holds the following lemma.

Lemma 2. *During the execution of PRA-TTC and the execution of the first phase of Filling-Gaps, for each $i \in N$ with $\overline{P}_i \neq \emptyset$, it holds that*

$$|R_i^a| = \sum_{j \in [m^*]} |R_{p_{i,j}}^p|.$$

Proof. Note that during the execution of PRA-TTC, if an agent i with $\overline{P}_i \neq \emptyset$ is assigned one submission due to the elimination of a cycle, then we know that one of her submissions that is incompletely assigned is also assigned to an agent that does not review it already. Hence, we see that until $\overline{P}_i$ becomes empty, we have that $|R_i^a| = \sum_{j \in [m^*]} |R_{p_{i,j}}^p|$.

Next, we focus on the execution of Filling-Gaps. We know that any $i \in N$ with $\overline{P}_i \neq \emptyset$ is included in U. In the first phase, the algorithm eliminates cycles in the greedy graph. With similar arguments as in the elimination of cycles in the preference graph, we get that during and after the first phase of Filling-Gaps,it is still true that $|R_i^a| = \sum_{j \in [m^*]} |R_{p_{i,j}}^p|$ for any $i \in U$ with $\overline{P}_i \neq \emptyset$. □

Next, we need to show that Line 8 of PRA-TTC is valid which is true if $|U| \leqslant k_p$.

Lemma 3. *PRA-TTC returns $|U| \leqslant k_p$.*

Proof. For each $i \in U$, from Lemma 2, we know that

$$|R_i^a| = \sum_{j \in [m^*]} |R_{p_{i,j}}^p| < m^* \cdot k_p \leqslant k_a$$

where the first inequality follows since there exists at least one submission of i that is assigned to less than k_p agents. Hence, we get that each $i \in U$ can review more submissions, when PRA-TTC terminates. Now, suppose for contradiction that at the last iteration of PRA-TTC, each agent $i \in U$ has an outgoing edge in the preference graph. In this case, we claim that there exists a directed cycle in the preference graph which is a contradiction since PRA-TTC would not have been terminated yet. To see that, note that each outgoing edge of an agent $i \in U$ either goes to another agent $i' \in U$, since i' can review more submissions, or goes to an agent $i' \notin U$ whose all submissions are completely assigned. In the latter case, i' has an outgoing edge to an agent in U by the definition of the preference graph. Thus, starting from any agent in U, we conclude in an agent in U and eventually we would found a cycle. Therefore, we have that there exists an agent $i^* \in U$ that at the last iteration of the algorithm arbitrary picks her incomplete submission p_{i^*,ℓ^*} and does not have any outgoing edge to any other agent. This means that all the agents that can review more submissions, already review p_{i^*,ℓ^*}. Since all the agents in $U \setminus \{i^*\}$ can review more submissions, we get that all of them are assigned p_{i^*,ℓ^*}. But since p_{i^*,ℓ^*} is not completely assigned, we conclude that $|U \setminus \{i^*\}| < k_p$, which means that $|U| \leqslant k_p$. □

We proceed by showing that Lines 11- 12 of Filling-Gaps are valid.

Lemma 4. *When Fillings-Gaps enter the second phase with a non empty U, for each $t \in [|U|]$ and for each $p_{\rho(t),\ell} \in \overline{P}_{\rho(t)}$, it exists a completely assigned submission $p_{i',\ell'}$ with $i' \in U \cup L \setminus \{\rho(t)\}$ that is not reviewed by $\rho(t)$, and it also exists an $i'' \in N$ that reviews $p_{i',\ell'}$, but she does not review $p_{\rho(t),\ell}$.*

Proof. When U is non empty and no more cycles exists in the greedy graph, the algorithm constructs the topological order of the greedy graph, denoted by $\vec{\rho}$.

First, we show the following proposition.

Proposition 1. *For each $t \in [|U|]$, $\rho(t)$ reviews all the incompletely assigned submissions of each $i \in U \setminus \{\rho(1), \dots, \rho(t-1), \rho(t)\}$.*

Proof. Since $\vec{\rho}$ is the topological ordering of the greedy graph, we have that no $i \in U \setminus \{\rho(1), \dots, \rho(t-1), \rho(t)\}$ has an outgoing edge to $\rho(t)$. But from Lemma 2, we get that $\rho(t)$ can review more submissions, since $\rho(t)$ has submissions that are incompletely assigned which means that

$$\sum_{\ell \in [m^*]} |R^p_{p_{\rho(1),\ell}}| < kp \cdot m^* \leqslant k_a.$$

Therefore, from the definition of the greedy graph, we get that $\rho(t)$ reviews all the incompletely assigned submissions of each i in $U \setminus \{\rho(1), \dots, \rho(t-1), \rho(t)\}$. □ (Proof of Proposition 1)

Next, we show by induction that for each $t \in [|U|]$ as long as $\overline{P}_{\rho(t)}$ is non-empty and it holds that $|R^a_{\rho(t)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(t),\ell}}|$, there exists a completely assigned submission $p_{i',\ell'}$ of an agent $i' \in U \cup L \setminus \{\rho(t)\}$ that is not reviewed by $\rho(t)$, and $i'' \in N$ that reviews $p_{i',\ell'}$ and does not review $p_{\rho(t),\ell} \in \overline{P}_{\rho(t)}$.

We start with $t = 1$. First, suppose for contradiction that $\rho(1)$ reviews all the submissions of all the agents in $U \cup L \setminus \{\rho(1)\}$. Then, we would have that

$$|R^a_{\rho(1)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(1),\ell}}| = k_p \cdot m^*,$$

where the first equality follows from the assumption that $|R^a_{\rho(1)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(1),\ell}}|$ and the second inequality follows from the facts that $|U \cup L \setminus \{\rho(1)\}| = k_p$ and each agent has m^* submissions. But then we would conclude that all the submissions of $\rho(1)$ are completely assigned since $\rho(1)$ has m^* submissions and each of them should be assigned to k_p reviewers which is a contradiction. Moreover, from Proposition 1 we know that $\rho(1)$ reviews all the incomplete assigned submissions that belongs to some agent $i \in U \setminus \{\rho(1)\}$. Hence, we get that since $\rho(1)$ reviews all the incompletely assigned submissions but she cannot review all the submissions of all agents in $i \in U \cup L \setminus \{\rho(1)\}$, there exists a completely assigned submission $p_{i',\ell'}$ that belongs to some $i' \in U \cup L \setminus \{\rho(1)\}$ and is not reviewed by $\rho(1)$. In addition, since $p_{i',\ell'}$ is reviewed by k_p agents and not from $\rho(1)$, while $p_{\rho(1),\ell} \in \overline{P}_{\rho(1)}$ is reviewed by strictly less than k_p agents, it exists an agent i'' that reviews the former submission but not the latter. It remains to show that during the execution of the 1-th iteration of the for loop in the second phase of Filling-Gaps, it holds that $|R^a_{\rho(1)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(1),\ell}}|$. Note that if every time that the algorithm enters the second while loop of the algorithm, this property is satisfied, then the property remains true at the end of this execution, since as we show above, in this case there are $p_{i',\ell'}$ and i'' with the desired properties, and therefore one incompletely assigned submission of $\rho(1)$ is assigned to a new reviewer and concurrently $\rho(1)$ is assigned a new submission to review. We get that $|R^a_{\rho(1)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(1),\ell}}|$ is true during the execution of the 1-st ieration of the for loop by noticing that from Lemma 2, we know that this is true when we first enter the while loop of the second phase.

Suppose that the hypothesis holds for $t - 1$. Note that from the base case and the hypothesis, at iteration t, all the submissions of each agent in $i' \in L \cup \{\rho(1), \dots, \rho(t-1)\}$ are completely assigned. Thus, any incompletely assigned submission, that does not belong to $\rho(t)$, belongs to some agent $i \in U \setminus \{\rho(1), \dots, \rho(t-1), \rho(t)\}$. But, from Proposition 1 we already know that $\rho(t)$ reviews any such submission. Moreover, we note that $\rho(t)$ cannot review all the submissions of all the agents in $U \cup L \setminus \{\rho(t)\}$. Indeed, if we assume for contradiction that $\rho(t)$ reviews all the submissions of all the agents in $U \cup L \setminus \{\rho(t)\}$, then we have that

$$|R^a_{\rho(t)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(t),\ell}}| = k_p \cdot m^*,$$

where the first inequality follows from the assumption that $|R^a_{\rho(t)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(t),\ell}}|$ and the second follows from the facts that $|U \cup L \setminus \{\rho(t)\}| = k_p$ and each of them has m^* submissions, which would imply that all the submissions of $\rho(t)$ are completely assigned. Hence, we get that since $\rho(t)$ reviews all the incompletely assigned submissions but cannot review all the submissions of all agents in $i \in U \cup L \setminus \{\rho(t)\}$, there exists a completely assigned submission that belongs to some $i' \in U \cup L \setminus \{\rho(t)\}$ and is not reviewed by $\rho(t)$. Moreover, we show that there exists i'' that reviews $p_{i',\ell'}$, but does not review $p_{\rho(t),\ell} \in \overline{P}_{\rho(t)}$. Indeed, since $p_{i',\ell'}$ is reviewed by k_p agents and not from $\rho(t)$, while $p_{\rho(t),\ell}$ is reviewed by strictly less than k_p agents, it exists an agent that reviews the former submission but not the latter. It remains to show that during the execution of the t-th iteration of the for loop in the second phase of Filling-Gaps, it holds that $|R^a_{\rho(t)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(t),\ell}}|$. Note that if when we first enter the while loop of the t-th iteration it is indeed true that $|R^a_{\rho(t)}| = \sum_{\ell \in [m^*]} |R^p_{p_{\rho(t),\ell}}|$, then during the whole execution of the while loop this property remains true, since as we show above, in this case there are $p_{i',\ell'}$ and i'' with the desired properties. From Lemma 2, we know that this is true when we enter the second phase of Filling-Gaps. Before, round t, if $\rho(t)$ is assigned a new submission to review, she is removed one of the old assigned submissions, while none of her incompletely assigned submissions is assigned to any agent. Hence, indeed we have the desired property, when we first enter the t-th round. □ (Proof of Lemma 4)

We are now ready to prove Lemma 1.

Proof of Lemma 1. We partition the agents in N, into two parts N_1 and N_2, where N_1 contains all the agents that do not belong in U that PRA-TTC returns and $N_2 = N \setminus N_1$. We proceed by separately showing that the assignment that CoBRA returns is valid over the agents in N_1 and over the agent in N_2.

Valid Assignment over the agents in N_1. Note that in PRA-TTC, each submission is assigned to at most k_p reviewers and therefore, during the execution of PRA-TTC, for each $i \in N$, it holds that $\sum_{j \in [m^*]} |R^p_{p_{i,j}}| \leqslant k_p \cdot m^*$. From Lemma 2, since $k_p \cdot m \leqslant k_a$, we get that in PRA-TTC, each $i \in N_1$ is not assigned more than k_a papers to review until the point where all of her submissions become completely assigned. After that point, an agent may still participate in a cycle as long as she reviews strictly less than k_a submissions. Therefore, when we exit PRA-TTC, each agent in N_1 does not review more than k_a submissions and all her submissions are completely assigned. In Filling-Gaps, from Lemma 4, we get that if an agent in N_1 is assigned a new submission to review, she is removed one of the submissions that she already reviews. Moreover, the assignments of submissions that belong to agents in N_1 do not change. Hence, we can conclude that the assignment that CoBRA returns is valid with respect to the agents in N_1.

Valid Assignment over the agents in N_2. From Lemma 2, we get that during the execution of PRA-TTC each agent i that is included in U that PRA-TTC returns reviews less than k_a submissions, since some of her submissions are not completely assigned (which means that $\sum_{\ell \in [m^*]} |R^p_{p_{i,\ell}}| < k_p \cdot m^*$). From the same lemma, we have that after the execution of the first phase of Filling-Gaps, it holds that $|R^a_i| = \sum_{\ell \in [m^*]} |R^p_{p_{i,\ell}}|$. Next, we show that this property remains true after the second phase of Filling-Gaps. Indeed, from Lemma 4, we have that during the second phase of Filling-Gaps, if $i \in U$ is assigned a new submission to review without one of her incompletely assigned submissions is assigned to a new reviewer, she is removed one of her assigned submissions; on the other hand, if one of her incompletely assigned submissions is assigned to a new reviewer, she is also assigned to review a new submission. Lastly, from Lemma 4, we conclude that in the second phase of Filling-Gaps, all the submissions become eventually completely assigned since in each iteration of the while loop, an incompletely assigned submission is assigned to one more reviewer. Therefore in the assignment that Filling-Gaps returns, no agent in N_2 reviews more than k_a submissions and all the submissions of the agents in N_2 are completely assigned, which means that the assignment is valid with respect to the agents in N_2 as well. □

B Supplementary Experiments

In Section 4, we show that TPMS and PR4A often motivate group of authors to deviate and redistribute their submissions among themselves. The size of a deviating groups is also an interesting measure, for evaluating if such a group indeed consists a distinct subcommunity of researchers that has incentives to build its own conference rather than an extremely tiny group of authors that could locally benefit by exchanging their papers for reviewing. In Table 3, we can see the maximum size of a successfully deviating coalition, averaged across 100 runs, together with the standard error. As before, each run

Dataset	Alg	USW	ESW
CVPR '17	CoBRA	1.644	0.000
	TPMS	1.970	0.000
	PR4A	1.919	0.384
CVPR '18	CoBRA	1.208	0.000
	TPMS	1.586	0.004
	PR4A	1.560	0.731
ICLR '18	CoBRA	0.251	0.015
	TPMS	0.284	0.038
	PR4A	0.278	0.086

Table 2: USW and ESW without subsampling on CVPR 2017 and 2018, and ICLR 2018.

Dataset	Alg	Largest Deviating Group
CVPR '17	TPMS	6.64 ± 0.77
	PR4A	7.50 ± 0.77
CVPR '18	TPMS	10.54 ± 1.29
	PR4A	11.49 ± 1.49
ICLR '18	TPMS	11.25 ± 1.76
	PR4A	15.01 ± 1.76

Table 3: Largest Size of Deviating Group on CVPR 2017 and 2018, and ICLR 2018.

is a subsampled dataset of size 100, so these can be interpreted as percentages. It seems that under both TPMS and PR4A across all three datasets, the largest deviating communities are 6-15$\%$ of the conference size, which we can indeed reflect the sizes of some of the largest subcommunities at CVPR and ICLR. Of course, there are deviating groups with smaller size as well which can consist smaller communities.