A SUPER* Algorithm to Optimize Paper Bidding in Peer Review

Tanner Fiez [1] **Nihar Shah** [2] **Lillian Ratliff** [1]

Abstract

A number of applications involve the sequential arrival of users, and require showing each user a set of items. It is well known that the order in which the items are presented to a user can have a significant impact on the performance of the system, due to phenomenon such as primacy effects. An example application (which forms the focus of this paper) is the bidding process in conference peer review. Here reviewers enter the system sequentially, each reviewer needs to be shown the list of submitted papers, and the reviewer then "bids" to review some papers. In deciding on the ordering of the list of papers to show, there are two competing goals: (i) satisfying reviewers by showing them relevant items, and (ii) obtaining sufficiently many bids for each paper to ensure high quality reviews for all papers. In this paper, we first develop a framework to study this problem in a principled manner. We then design an algorithm called SUPER*, inspired by the A* algorithm, for this goal. We show that SUPER* has a number of attractive properties (some analogous to that of A*). Finally, we conduct experiments which reveal that our algorithm is quite robust to the complexities arising in the real world.

1. Introduction

It is well-known that peer review is an essential process for ensuring the quality and scientific value of research (Black et al., 1998; Thurner & Hanel, 2011). One of the more daunting challenges in peer review is matching or assigning papers to qualified and willing reviewers. Bidding has emerged as a mechanism for aiding in and improving this process under the guise that engagement of the reviewer leads to assignments more aligned with their preferences and hence, enhanced review quality (Di Mauro et al., 2005).

[1]Department of Electrical and Computer Engineering, University of Washington [2]Machine Learning and Computer Science Departments, Carnegie Mellon University. Correspondence to: Tanner Fiez <fiezt@uw.edu>.

Real-World Sequential Decision Making Workshop at ICML 2019. Copyright 2019 by the author(s).

In typical peer review process, when the bidding phase begins, reviewers enter the system in an arbitrary sequential order. Upon entering, a list of papers is shown and the reviewer places bids on papers they would prefer to review.

It is known that the order of papers presented to reviewers in the bidding stage can greatly impact the number of bids that a paper receives (Cabanac & Preuss, 2013). From the perspective of the platform, there are two competing goals: (i) ensure that each paper has a sufficient number of bids, and (ii) ensure individual reviewer satisfaction by showing relevant papers. With regard to (i), the platform aims to choose a display order for each reviewer such that following the bidding process, each paper has at least a certain number of bids. The main objective of ensuring a minimum number of bids on each paper is to improve review quality for all papers (Shah et al., 2018). The well-documented primacy effect suggests that papers shown on top of the ordering are the ones on which reviewers are more likely to bid. This objective, strongly suggests that papers with few bids should be placed relatively high in the list.

With regard to (ii), the platform aims to choose an ordering of papers to display that are 'well-matched' to the particular reviewer. That is, the set of papers to be displayed is the set of papers on which the particular reviewer is most likely to bid. There are a number of reasons to select well-matched papers: for instance, showing poorly matched papers at the top may cause the reviewer to opt-out and disengage with the system even if further down the list there was an option on which they would have happily bid. In particular, a poorly selected ordering may result in a significantly lower overall bid quantity from a particular reviewer.

There is a limited amount of fundamental research on the problem of optimizing the display order during the bidding process, and much less so in consideration of both of the objectives identified above. In practice, the display order is typically determined via heuristics such as a predetermined fixed ordering (e.g., order of submission), ordering by number of bids, or ordering by 'similarity scores'.

Poor bidding can in fact have a significant negative impact on the overall quality of the review process (Rodriguez et al., 2007). One of the key reasons that bidding can fail is that papers are suboptimally displayed to the reviewers. Consider a paper that is not an ideal match for any of the reviewers in the system. If papers are ranked for display simply by how well matched they are to reviewers (e.g., by commonly used similarity score), this particular paper may be very low in the ranking and hence, not receive many, if any, bids. For instance, a recent study from (Shah et al., 2018) conducted on the Neural Information Processing Systems (NeurIPS) 2016 conference revealed that 1090 papers (out of the 2000 used in the study) had no positive bids at all.

On the other hand, if papers are inversely ranked by the current number of bids they have, then papers with low bids are more likely to be higher up on the list regardless of how well-matched they are to a particularly reviewer. This display order may cause reviewer dissatisfaction, which in the worst case could result in zero bids.

Similarly, ordering heuristics that are based on a fixed baseline may lead to bias in the review process. Indeed, in the report of a study of 42 peer-reviewed conferences in Computer Science, it was observed that under a fixed ordering, the number of bids decreases with relative order of submission (Cabanac & Preuss, 2013). It was concluded that the later the paper is submitted, the less bids it will receive.

Given the shortcomings of existing peer review bidding systems, we study the important problem of **selecting the ordering of papers to display to each arriving reviewer** from a fundamental perspective. We begin with a formal problem description in the next section, and then move on to present our main results.

2. Problem Statement

Let d denote the number of papers and n denote the number of reviewers. For each reviewer-paper pair, we have access to a *similarity score* that captures the similarity between the reviewer and the paper. We use the notation $S_{i,j} \in [0,1]$ to denote the given similarity between any reviewer $i \in [n]$ and paper $j \in [d]$. A higher value of the similarity indicates a greater relevance of the paper to that reviewer. There are various systems (Price et al., 2010; Charlin & Zemel, 2013) in use today that compute the similarities, and in our work, we will treat them as being given.

In the bidding period, reviewers sequentially arrive into the system. The goal is to determine the ordering of the papers to show to a reviewer on arrival. We assume for simplicity that a reviewer arrives only after the previous reviewer has completed their bidding. We do not make any assumptions on the order of arrival of the reviewers.

Let Π_d denote the set of all possible $d!$ permutations of d papers. In what follows, for any reviewer $i \in [n]$, we let $\pi_i \in \Pi_d$ denote the ordering of the papers shown to reviewer i. We additionally use the notation $\pi_i(j)$ to denote the position of paper $j \in [d]$ in the ordering π_i. When deciding the ordering, one has access to the actual bids by all reviewers who arrived in the past (but does not have access to the bids made by the current or future reviewers). In this setting, the goal is to optimize the following objective.

Objective (gain function). Any algorithm to determine the ordering of papers must trade-off between two competing objectives: ensuring that each paper receives at least a reasonable number of bids and ensuring that each reviewer gets to see relevant papers at the top. A combination of these two objectives will comprise our "gain function," which is the objective we aim to optimize. We first separately discuss the two competing objectives.

Paper-side objective: The paper-side objective is associated with a given function $\gamma_p : \{0, \ldots, n\} \to \mathbb{R}$. At the end of the entire bidding procedure, the paper-side objective is

$$\mathcal{G}^p = \textstyle\sum_{j\in[d]} \gamma_p(g_j),$$

where g_j is the number of bids received by paper j. We assume that the function γ_p is non-decreasing and concave. The non-decreasing property represents an improved gain if there are more bids, and the concavity property captures diminishing returns.

While our algorithm handles general functions γ_p, we discuss two examples here for concreteness. One choice, associated with a given parameter $r \geq 1$, is $\gamma_p(x) = \min\{x, r\}$ and indicates that the program chairs emphasize having at least r bids per paper, but fewer bids also provide utility. A second example is $\gamma_p(x) = \sqrt{x}$, which is a smooth function of x and also captures the diminishing returns on bids.

Reviewer-side objective: In the bidding process, we wish to ensure that reviewers are shown papers with high relevance at the top. For any reviewer $i \in [n]$, we let $\pi_i \in \Pi_d$ denote the ordering (permutation) of the papers shown to the reviewer. Let $S_{i,:} \in [0,1]^d$ denote a d-length vector representing the similarities of reviewer i with the d papers. The reviewer-side gain is associated with some pre-specified function $\gamma_r : [d] \times [0,1] \to \mathbb{R}$ and defined by

$$\mathcal{G}^r = \textstyle\sum_{i\in[n]} \sum_{j\in[d]} \gamma_r(\pi_i(j), S_{i,j}).$$

The function γ_r is assumed to be continuous, as well as non-increasing in the position and non-decreasing in the similarity to reflect the objective.

There are several suitable choices for the function γ_r. A function commonly found in data mining, known as Discounted Cumulative Gain (DCG) (Järvelin & Kekäläinen, 2000), which we consider for our application, is given by:

$$\gamma_r(\pi_i(j), S_{i,j}) = \frac{2^{S_{i,j}} - 1}{\log_2(\pi_i(j) + 1)},$$

where we have set the "relevance" parameter in DCG to be the similarity $S_{i,j}$.

Overall gain function: Finally, we assume there is a hyperparameter $\lambda \geq 0$, chosen by the program chairs, which trades off between these two objectives:

$$\mathbb{E}[\mathcal{G}] = \mathbb{E}\big[\mathcal{G}^p + \lambda \mathcal{G}^r\big],$$

where the expectation is taken over the randomness in the bids made by the reviewers and any randomness in the algorithm. The goal is to determine the orderings of papers to show to each reviewer in order to maximize the expected value of the overall gain. The expectation is necessary since the bids by reviewers are stochastic, as explained below.

Reviewer bidding model. An important aspect of any system that displays a list is the presence of primacy effects. In the context of our problem setting, the primacy effect means a reviewer is more likely to bid on a paper shown at the top of the list rather than much later. We capture this using a pre-specified non-increasing function $f : \{1, \ldots, d\} \to [0, 1]$, where $f(x)$ captures the primacy effect at position x. A second aspect of bidding is that a reviewer is more likely to bid on papers with greater similarity, although the reviewer may not bid on exactly the highest similarity papers since the similarities are noisy. We now encapsulate both these aspects by assuming each reviewer i bids on papers independently with probability

$$\mathbb{P}(\text{reviewer } i \text{ bids on paper } j) = S_{i,j} \times f(\pi_i(j)) \; \forall \, j \in [d].$$

Baselines. The following are three baselines to which we compare. These baselines are used commonly in practice.

Random baseline (**RAND**): A commonplace practice is to show the papers to reviewers in some fixed order, such as, in order of submission of the papers. We consider a variant of this practice, in which each reviewer is shown a randomly selected paper ordering.

Similarity baseline (**SIM**): A second commonplace practice is to order the papers according to their similarities. In other words, any reviewer $i \in [n]$ is shown the papers in order of the values in $S_{i,:}$ (where the paper with maximum similarity is shown at the top, and so on). Any ties are broken by showing papers with fewer bids higher, and further ties are broken uniformly at random.

Bid baseline (**BID**): A third baseline shows papers in order to greedily optimize the minimum bids. Each reviewer is shown papers in increasing order of bids received so far (from the reviewers who arrived before this reviewer). Any ties are broken in favor of the paper with a higher similarity, and further ties are broken uniformly at random.

3. SUPER* Algorithm

The key challenge in designing a suitable algorithm for the problem at hand stems from the fact that the paper-side gain function is coupled between the orderings of papers presented to all reviewers and cannot be realized until the entire bidding process is complete. Conversely, the reviewer-side gain is decoupled across reviewers so that the choices of an algorithm prior to and after a reviewer arrives do not alter the reviewer-side gain that can be obtained from any given reviewer. Thus, an algorithm for this problem is required to make local decisions, where the effect of the decision on the global gain (or cost) is only partially known. This perspective is reminiscent of the A* algorithm (Hart et al., 1968), and using A* as an inspiration, we now present an algorithm which we call SUPER* for our problem[1].

The A* algorithm operates with a goal of finding the minimum cost path between a pair of vertices in a cost-weighted graph. For any node in consideration, it considers two functions: a function which captures the cost so far and a second function—called the "heuristic"—which captures some estimate of the cost from the current node to the destination. The A* algorithm then finds a path based on these two functions. Before moving to a description of SUPER*, we first discuss the heuristic in the context of the problem at hand.

3.1. Heuristic for future bids.

In a manner analogous to the A* algorithm, at any point in time SUPER* keeps track of the gains so far and also takes as input a heuristic that captures the "unseen" events. The heuristic in A* provides, for every vertex in the given graph, an estimate of the cost incurred in the future. Analogously, the heuristic in SUPER* provides, for every arrival of a reviewer, an estimate of the number of bids each paper will receive in the future. Formally, let us index the reviewers as $1, \ldots, n$ in the order of their arrival (note that this order is unknown a priori). The heuristic comprises a collection of vectors $h_1, \ldots, h_n$, where each $h_i \in \{0, \ldots, n\}^d$ represents an estimate of the number of bids each of the d papers will receive from all future reviewers $\{i+1, \ldots, n\}$. The vector h_i is provided to the SUPER* algorithm on arrival of the i^{th} reviewer. Two examples of heuristic functions that we consider in the subsequent narrative are defined as follows:

- *Zero heuristic:* $h_i = 0$ for every $i \in [n]$.
- *Mean heuristic:* This function computes the expected number of bids each paper will receive if the permutations shown to all future reviewers are chosen independently and uniformly at random. Formally, for every $i \in [n-1]$, the mean heuristic is defined as $h_i = f_0 \sum_{i'=i+1}^{n} S_{i',:}$ where $f_0 = \frac{1}{d} \sum_{j' \in [d]} f(j')$.

We set $h_n = 0$, implying that there will be no more bids on any paper after the last reviewer. This is analogous to setting the heuristic value to zero for the target vertex in the

[1]The name SUPER* stands for SUperior PERmutations and also indicates the inspiration from A*.

```
Algorithm 1: SUPER*
Input: $\gamma_p : \{0,\dots,n\} \to \mathbb{R}$, paper-side gain function
       $\gamma_r : [d] \times [0,1]^d \to \mathbb{R}$, reviewer-side gain
       $f : [d] \to [0,1]$, bidding model
       $\lambda \geq 0$, hyper-parameter on reviewer-side gain
       $S \in [0,1]^{n \times d}$, similarity matrix.
Algorithm:
  1. Initialize bids on each paper to zero: $g_1 \leftarrow 0_d$
  2. For each reviewer arrival $i \in [n]$
     (a) Compute or take input heuristic $h_i \in \{0,\dots,n\}^d$
     (b) $\pi_i \leftarrow$ FindPaperOrder
     (c) Present papers to reviewer in an order following $\pi_i$
     (d) Observe vector of bids $b_i \in \{0,1\}^d$
     (e) Update paper bid counts: $g_{i+1} \leftarrow g_i + b_i$
```

```
Algorithm 2: FindPaperOrder
  1. Compute weight matrix $w \in \mathbb{R}^{d\times d}$ such that
     $w_{j,k} \leftarrow \lambda\gamma_r(k, S_{i,j})$
       $+ S_{i,j} f(k)(\gamma_p(g_{i-1,j} + h_{i,j} + 1) - \gamma_p(g_{i-1,j} + h_{i,j}))$
  2. Solve linear program to obtain $x^* \in \mathbb{R}^{d\times d}$:
     $x^* = \arg\max_{x\in[0,1]^{d\times d}} \sum_{j\in[d]}\sum_{k\in[d]} w_{j,k}x_{j,k}$
       s.t. $\sum_{k\in[d]} x_{j,k} = 1 \ \forall\, j \in [d], \quad \sum_{j\in[d]} x_{j,k} = 1 \ \forall\, k \in [d].$
  3. $\pi_i(j) \leftarrow k$ such that $x^*_{j,k} = 1$ for each $j \in [d]$
  Output: $\pi_i$
```

A* algorithm.

3.2. Intuition behind the algorithm

We first provide some intuition about the SUPER* algorithm, and subsequently present a formal description.

Since a primary impediment to designing an algorithm is the inability to realize the paper-side gain until the end of the bidding process, we begin by considering the setting when $(n-1)$ reviewers have already departed, and the problem is to determine the ordering of papers shown to the final reviewer. In this scenario, we have access to the bids of all $(n-1)$ reviewers that have already arrived and the orderings of papers presented to them. We use the notation $g_{n,j} \in \{0,\dots,n\}$ to denote the number of bids received by any paper $j \in [d]$ at the time of arrival of the last reviewer. The values $g_{n,1},\dots,g_{n,d}$ are thus known at the time when the final reviewer arrives. As a result, we can formulate an optimization problem for the final reviewer n to maximize the gain (2) in the following manner. For every $j \in [d]$, let $\mathcal{B}_{n,j}$ denote a Bernoulli random variable with mean $S_{n,j}f(\pi_n(j))$ independent of all else. The random variable $\mathcal{B}_{n,j}$ represents the bid of the final reviewer on paper j. The optimization problem can be written as:

$$\arg\max_{\pi_n\in\Pi_d} \quad \textstyle\sum_{j\in[d]} \mathbb{E}[\gamma_p(g_{n,j} + \mathcal{B}_{n,j})] + \lambda \textstyle\sum_{j\in[d]} \gamma_r(\pi_n(j), S_{n,j}) \tag{1}$$

where the expectation is taken over the distribution of the random variables $\mathcal{B}_{n,1},\dots,\mathcal{B}_{n,d}$.

Observe that the constraint set for the optimization problem in (1) is the set Π_d of all permutations. This set is, in general, not very well behaved (Ailon et al., 2008; Shah et al., 2016), which makes even this one-step optimization a challenge. As we discuss later (in Theorem 1 and its proof), SUPER* for the final reviewer optimally solves (1) in a computationally efficient manner. The aforementioned subproblem forms the starting point for the SUPER* algorithm. Now that we know to handle a single (last) reviewer in an optimal fashion, we now describe the SUPER* algorithm for a general reviewer, say, $i \in [n]$. When reviewer i arrives, we have access to the number of bids made by all past reviewers on any paper j, which we denote by $g_{i,j} \in \{0,\dots,i-1\}$.

We now recall the A* algorithm: for any vertex, A* considers the cost "g" so far and a heuristic estimate "h" of the cost-to-go. Then, considering the cost of any vertex as "$g+h$", the A* algorithm takes the action which is the one-step optimal action (i.e., chooses the neighboring vertex with the smallest value of "$g+h$").

In an analogous fashion, SUPER* considers the number of bids so far (g_i) and takes as input a heuristic (h_i) for the number of bids in the future. Then, considering the number of bids from all other reviewers as "$g_i + h_i$", the SUPER* algorithm takes the action which is the one-step optimal action. In other words, SUPER* solves for each paper ordering using:

$$\arg\max_{\pi_i\in\Pi_d} \sum_{j\in[d]} \mathbb{E}[\gamma_p(g_{i,j} + h_{i,j} + \mathcal{B}_{i,j})] + \lambda \sum_{j\in[d]} \gamma_r(\pi_i(j), S_{i,j}) \tag{2}$$

where $\mathcal{B}_{i,j}$ is a Bernoulli random variable with mean $S_{i,j}f(\pi_i(j))$ independent of all else.

3.3. SUPER* formal description

The SUPER* algorithm is in Algorithm 1. Each time a reviewer arrives, SUPER* calls to a routine, FindPaperOrder (Algorithm 2), which uses the information available to SUPER* and determines an ordering of papers to show.

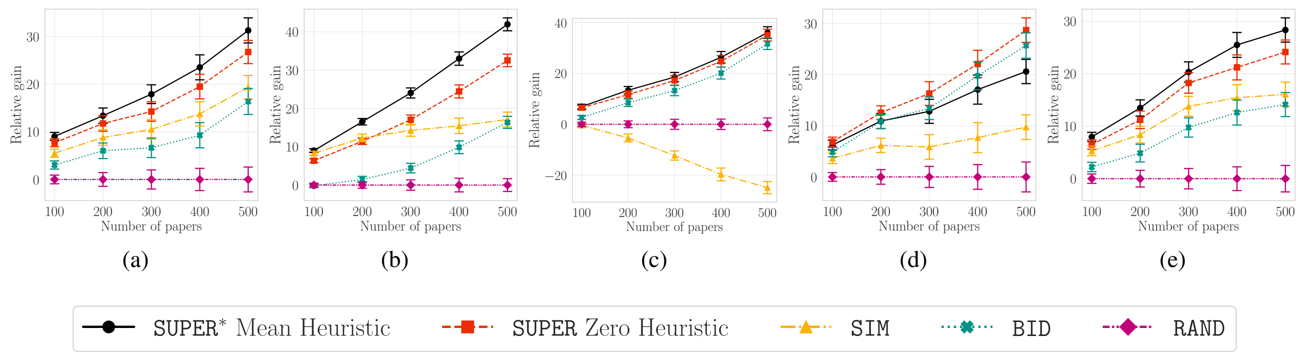

Figure 1. (a) Model holds and $\gamma_p(x) = \sqrt{x}$; (b) Model holds and $\gamma_p(x) = \min\{x, 3\}$; (c) $f(x)$ is assumed to be $\frac{1}{\log_2(x+1)}$ but instead it is $f(x) = \frac{1}{\sqrt{x}}$; (d) $n/2$ reviewers out of the n reviewers arrive to bid; (e) Poisson(1) reviewers arrive at each discrete time interval. The paper-side gain function in (c–e) is $\gamma_p(x) = \sqrt{x}$. The gain reported in the figures is relative to the RAND baseline.

In the proof of Theorem 1, we show the problem in (1) can equivalently be formulated as the linear program with a totally unimodular constraint set. As a result, the optimal solution is integer valued. In fact, the optimization problem is in the form of the linear assignment problem and the optimal solution can be obtained with a time complexity of $\mathcal{O}(d^3)$ using the Hungarian algorithm. The FindPaperOrder subprocedure exactly solves this problem. In Appendix B, we discuss conditions under which a solution can be obtained in a more efficient manner from a simple sorting routine that requires a time complexity of only $\mathcal{O}(d \log(d))$.

4. Theoretical Results

SUPER* has a number of attractive theoretical properties, including some analogous to the A* algorithm. We discuss one such property ("local optimality") here and relegate the rest to a future longer version of the paper.

The property of local optimality, as the name suggests, means that the algorithm is optimal with respect to the reviewer under consideration. Achieving even a good local performance in a computationally efficient manner is challenging due to the optimization over permutations in (1). The following result shows that SUPER*, which is computationally efficient, is indeed locally optimal. The result is presented in terms of the final reviewer for simplicity.

Theorem 1 (Local optimality)**.** *SUPER*, with any heuristic, is optimal for the final reviewer.*

The proof of Theorem 1 is provided in Appendix A.

5. Experimental Results

In our simulations, we show the empirical results of the SUPER* algorithm with a zero and mean heuristics, along the baselines described in Section 2. A consistent procedure is performed to evaluate performance. Given a class of similarity matrix, we sample a similarity matrix and shuffle the rows of the matrix so that the order of reviewers is arbitrary and then simulate each algorithm using this similarity matrix and record the gains obtained. We repeat the process 50 times and track the mean gains and standard error for each algorithm. We present figures showing the relative mean gain of each algorithm to the mean gain of the RAND baseline so it is clear how the relative performance scales.

For each simulation we consider a DCG reviewer-side gain function and a bidding function of $f(x) = \frac{1}{\log_2(x+1)}$. The hyper-parameter λ is selected so that each component of the gain function is nearly equal. The similarity matrix class we consider is matrices of rank 10 generated from summing rank-1 matrices for which the vectors are drawn from a Beta distribution with parameters $\alpha = 5, \beta = 2$. We fix the number of reviewers to be 100 and sweep the paper count.

We begin showing simulations for situations that faithfully reflect our model. In Fig. 1a the paper-side gain function is $\gamma_p(x) = \sqrt{x}$ and in Fig. 1b the paper-side gain function is $\gamma_p(x) = \min\{x, 3\}$. For the remaining simulations, the paper-side gain function is $\gamma_p(x) = \sqrt{x}$. In Fig. 1c we show an example where the assumed bidding model is $f(x) = \frac{1}{\log_2(x+1)}$, but the underlying bidding model is instead $f(x) = \frac{1}{\sqrt{x}}$. Fig. 1d shows an example where only half of the reviewers arrive, and the algorithms do not have knowledge of this event. Finally, in Fig. 1e we show an example where Poisson(1) reviewers arrive at once, and each algorithm must present paper orderings to each at once.

The simulations show our proposed algorithm performs well when the model holds, and can be robust to realistic scenarios where the model fails to hold. While we only show results for a certain similarity matrix class, we observe qualitatively similar results more generally.

References

Ailon, N., Charikar, M., and Newman, A. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23, 2008.

Black, N., Van Rooyen, S., Godlee, F., Smith, R., and Evans, S. What makes a good reviewer and a good review for a general medical journal? *Jama*, 280(3):231–233, 1998.

Cabanac, G. and Preuss, T. Capitalizing on order effects in the bids of peer-reviewed conferences to secure reviews by expert referees. *Journal of the American Society for Information Science and Technology*, 64(2):405–415, 2013.

Charlin, L. and Zemel, R. The toronto paper matching system: an automated paper-reviewer assignment system. 2013.

Di Mauro, N., Basile, T. M., and Ferilli, S. Grape: An expert review assignment component for scientific conference management systems. In *International conference on industrial, engineering and other applications of applied intelligent systems*, pp. 789–798. Springer, 2005.

Diaconis, P. and Graham, R. L. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):262–268, 1977.

Hart, P. E., Nilsson, N. J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

Järvelin, K. and Kekäläinen, J. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 41–48. ACM, 2000.

Price, S., Flach, P. A., and Spiegler, S. Subsift: a novel application of the vector space model to support the academic research process. In *Proceedings of the First Workshop on Applications of Pattern Analysis*, pp. 20–27, 2010.

Rodriguez, M. A., Bollen, J., and Van de Sompel, H. Mapping the bid behavior of conference referees. *Journal of Informetrics*, 1(1):68–82, 2007.

Shah, N., Balakrishnan, S., Guntuboyina, A., and Wainwright, M. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*, pp. 11–20, 2016.

Shah, N. B., Tabibian, B., Muandet, K., Guyon, I., and Von Luxburg, U. Design and analysis of the nips 2016 review process. *The Journal of Machine Learning Research*, 19(1):1913–1946, 2018.

Thurner, S. and Hanel, R. Peer-review in a world with rational scientists: Toward selection of the average. *The European Physical Journal B*, 84(4):707–711, 2011.

A. Proof of Theorem 1

Proof. We consider optimizing the objective to determine the ordering of papers to present a reviewer, given the history of observable information and a heuristic. In this proof, we show that selecting the optimal ordering to present to the final reviewer can be reduced to an optimization problem that can be solved efficiently. Since the `SUPER`* algorithm solves this optimization problem to determine an ordering of papers to present the final reviewer, it is an optimal algorithm for the final reviewer.

We now formally define the history available to an algorithm when a reviewer arrives.

Definition 1 (History). *The history available to an algorithm when a reviewer arrives is defined as the set containing the orderings presented and the bids obtained from each reviewer previously. The history set available when reviewer $i \in [n]$ arrives is defined as $\mathcal{H}_{i-1} = \{\pi_s, b_s : s = 1, \dots, i-1\}$ where π_s is the paper ordering presented to reviewer s and b_s contains the bid realizations on each paper. Observe that the number of bids on each paper when a reviewer arrives is deterministic given the history.*

The optimization problem for the final reviewer n is

$$\begin{aligned} \arg\max_{\pi_n} \quad & \sum_{j\in[d]} \mathbb{E}[\gamma_p(g_j)|\mathcal{H}_{n-1}] + \lambda \sum_{i\in[n]} \sum_{j\in[d]} \mathbb{E}[\gamma_r(\pi_i(j), S_{i,j})|\mathcal{H}_{n-1}] \\ \text{s.t.} \quad & \pi_n \in \Pi_d, \end{aligned} \tag{3}$$

where the expectation is taken over the randomness in the bid to be placed by the reviewer. Equivalently, the optimization problem is defined as

$$\begin{aligned} \arg\max_{\pi_n} \quad & \sum_{j\in[d]} \mathbb{E}[\gamma_p(g_{n-1,j} + \mathcal{B}_{n,j})] + \lambda \sum_{j\in[d]} \gamma_r(\pi_n(j), S_{n,j}) \\ \text{s.t.} \quad & \pi_n \in \Pi_d. \end{aligned}$$

This representation follows from simplifying the expression and removing terms that do not depend on the optimization variable. The number of bids on paper $j \in [d]$ is expressed as the sum of the deterministic number of bids on the paper prior to the final reviewer denoted as $g_{i-1,j}$ and a Bernoulli random variable $\mathcal{B}_{n,j}$ with parameter $S_{n,j} f(\pi_n(j))$ representing the random bid of the final reviewer on the paper.

We can simplify the paper-side loss term by expanding the expectation for each $j \in [d]$. Observe that

$$\gamma_p(g_{n-1,j} + \mathcal{B}_{n,j}) = \begin{cases} \gamma_p(g_{n-1,j} + 1) & w.p. \quad S_{n,j} f(\pi_n(j)) \\ \gamma_p(g_{n-1,j}) & w.p. \quad 1 - S_{n,j} f(\pi_n(j)) \end{cases}.$$

Hence,

$$\begin{aligned} \mathbb{E}[\gamma_p(g_{n-1,j} + \mathcal{B}_{n,j})] &= S_{n,j} f(\pi_n(j)) \gamma_p(g_{n-1,j} + 1) + (1 - S_{n,j} f(\pi_n(j))) \gamma_p(g_{n-1,j}) \\ &= S_{n,j} f(\pi_n(j)) (\gamma_p(g_{n-1,j} + 1) - \gamma_p(g_{n-1,j})) + \gamma_p(g_{n-1,j}). \end{aligned}$$

Dropping the term that does not depend on the optimization variable gives the problem

$$\begin{aligned} \arg\max_{\pi_n} \quad & \sum_{j\in[d]} S_{n,j} f(\pi_n(j)) (\gamma_p(g_{n-1,j} + 1) - \gamma_p(g_{n-1,j})) + \lambda \sum_{j\in[d]} \gamma_r(\pi_n(j), S_{n,j}) \\ \text{s.t.} \quad & \pi_n \in \Pi_d. \end{aligned} \tag{4}$$

We now reformulate the problem into an integer programming problem as follows:

$$\begin{aligned} \arg\max_{x\in\mathbb{R}^{d\times d}} \quad & \sum_{j\in[d]} \sum_{k\in[d]} S_{n,j} f(k) (\gamma_p(g_{n-1,j} + 1) - \gamma_p(g_{n-1,j})) \cdot x_{j,k} + \lambda \sum_{j\in[d]} \sum_{k\in[d]} \gamma_r(k, S_{n,j}) \cdot x_{j,k} \\ \text{s.t.} \quad & \sum_{k\in[d]} x_{j,k} = 1 \ \forall \, j \in [d] \\ & \sum_{j\in[d]} x_{j,k} = 1 \ \forall \, k \in [d] \\ & x_{j,k} \in \{0, 1\} \ \forall \, j, k \in [d]. \end{aligned}$$

In this formulation x is a $d \times d$ matrix for which $x_{j,k}$ is an indicator of paper $j \in [d]$ being shown at position $k \in [d]$. The constraint $\sum_{k\in[d]} x_{j,k} = 1 \; \forall \; j \in [d]$ ensures each paper is included strictly once in the ordering shown to the reviewer. The constraint $\sum_{j\in[d]} x_{j,k} = 1 \; \forall \; k \in [d]$ ensures strictly one paper is selected to be shown at each position. The final constraint ensures that each index of x is integer valued. Since the constraint matrix is totally unimodular, we can simplify the final constraint to be $0 \leq x_{j,k} \leq 1 \; \forall \; j, k \in [d]$ and the optimal solution will be the integral optimal solution. For notational purposes, we now define weights

$$w_{j,k} = S_{n,j} f(k)(\gamma_p(g_{n-1,j} + 1) - \gamma_p(g_{n-1,j})) + \lambda\gamma_r(k, S_{n,j}) \; \forall \; j, k \in [d].$$

The optimization problem arising from recognizing that the integer constraint can be relaxed and substituting the weights we defined is as follows

$$\begin{aligned} x^* = \arg\max_{x\in\mathbb{R}^{d\times d}} \quad & \sum_{j\in[d]}\sum_{k\in[d]} w_{j,k} x_{j,k} \\ \text{s.t.} \quad & \sum_{k\in[d]} x_{j,k} = 1 \; \forall \; j \in [d] \\ & \sum_{j\in[d]} x_{j,k} = 1 \; \forall \; k \in [d] \\ & 0 \leq x_{j,k} \leq 1 \; \forall \; j, k \in [d]. \end{aligned}$$

This optimization problem is in the form of the linear assignment problem. As a result, using the Hungarian algorithm the optimal solution can be obtained with a time complexity of $\mathcal{O}(d^3)$. The position paper $j \in [d]$ is shown to reviewer $i \in [n]$ is extracted from x^* using the relation $\pi_i(j) = k$ such that $x^*_{j,k} = 1$.

Optimality of SUPER*. The SUPER* algorithm solves the problem given above for the final reviewer with any heuristic since $h_n = 0$. As a result, it is optimal for the final reviewer.

□

B. Computationally More Efficient SUPER* in Certain Settings

We now consider when it is feasible to obtain a problem formulation that can be solved with improved time complexity under certain conditions. The problem in (3) can be formulated in a favorable way when the reviewer-side gain function γ_r and the bidding model function f are of an equivalent form. Precisely, when each function is of a fractional form with the dependence on the position a paper is shown only appearing in the denominators, we can obtain the following optimization problem directly from (4):

$$\begin{aligned} \arg\max_{\pi_i} \quad & \sum_{j\in[d]} \alpha_{i,j} f(\pi_i(j)) \\ \text{s.t.} \quad & \pi_n \in \Pi_d, \end{aligned} \tag{5}$$

where

$$\alpha_{i,j} = S_{i,j}(\gamma_p(g_{i-1,j} + h_{i,j} + 1) - \gamma_p(g_{i-1,j} + h_{i,j})) + \lambda\tilde{\gamma_r}(S_{i,j}) \; \forall \; j \in [d],$$

and $\tilde{\gamma_r}$ is a function only including the numerator of γ_r and no longer depending on the paper ordering that is presented. We can also consider this formulation when the hyper-parameter on the reviewer-side gain function is set to zero so that reviewer-side gain function no longer factors into the objective.

The optimal solution to (5) is simply to order the papers in decreasing order of their corresponding values of $\alpha_{i,j}$ and, as a result, requires a time complexity of just $\mathcal{O}(d\log(d))$. To be explicit, we obtain the solution using the relation $\pi_i = r(\alpha_i)$, where $r : \mathbb{R}^d \to [d]^d$ is a function that returns the rank from maximum to minimum of each input in place given a vector of ordered inputs.