Advances at the Interface of Combinatorial Optimization and Computations Social Choice: Mathematical Formulations, Structural Decompositions, and Analytical Insights

by

Sina Akbari

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved September 2022 by the
Graduate Supervisory Committee:

Adolfo R. Escobedo, Chair
Geunyeong Byeon
Jorge Sefair
Shin-yi Wu

ARIZONA STATE UNIVERSITY

December 2022

ABSTRACT

The rank aggregation problem has ubiquitous applications in operations research, artificial intelligence, computational social choice, and various other fields. Generally, rank aggregation is utilized whenever a set of judges (human or non-human) express their preferences over a set of items, and it is necessary to find a consensus ranking that best represents these preferences collectively. Many real-world instances of this problem involve a very large number of items, include ties, and/or contain partial information, which brings a challenge to decision-makers. This work makes several contributions to overcoming these challenges.

Most attention on this problem has focused on an NP-hard distance-based variant known as Kemeny aggregation, for which solution approaches with provable guarantees that can handle difficult large-scale instances remain elusive. Firstly, this work introduces exact and approximate methodologies inspired by the social choice foundations of the problem, namely the Condorcet criterion, to decompose the problem. To deal with instances where exact partitioning does not yield many subsets, it proposes Approximate Condorcet Partitioning, which is a scalable solution technique capable of handling large-scale instances while providing provable guarantees.

Secondly, this work delves into the rank aggregation problem under the generalized Kendall-tau distance, which contains Kemeny aggregation as a special case. This new problem provides a robust and highly-flexible framework for handling ties. First, it derives exact and heuristic solution methods for the generalized problem. Second, it introduces a novel social choice property that encloses existing variations of the Condorcet criterion as special cases.

Thirdly, this work focuses on top-k list aggregation. Top-k lists are a special form of item orderings wherein out of n total items only a small number of them, k, are explicitly ordered. Top-k lists are being increasingly utilized in various fields includ-

ing recommendation systems, information retrieval, and machine learning. This work introduces exact and inexact methods for consolidating a collection of heterogeneous top- lists. Furthermore, the strength of the proposed exact formulations is analyzed from a polyhedral point of view. Finally, this work identifies the top-100 U.S. universities by consolidating four prominent university rankings to assess the computational implications of this problem.

I dedicate this work to my beloved family, my mother Farangis, my late father Masoud, my sisters Maryam and Samira, and my brother Saeed

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Adolfo R. Escobedo, for his immense support, help, and guidance. He had a profound impact on my professional and personal growth, and helped me to become a better researcher and a better person. He was very compassionate, understating, and helpful. I am grateful for having him as my advisor. Without him this dissertation would not exist.

I would like to thank Dr. Jorge Sefair, Dr. Geunyeong Byeon, and Dr. Shin-yi WU for taking part in my dissertation committee and providing comments and guidance that improved this work.

Finally, I would like to thank my family for their irreplaceable and invaluable love and support.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Motivation and Overview

Rank aggregation is a well-studied problem in operations research, computer science, and computational social choice, which arises in a variety of situations where m judges (i.e., voters) are asked to rank n items (i.e., candidates) based on some specific quality of interest; effectively, the rankings submitted by the judges' can be considered as their ballots in an election. The objective is to find a *consensus ranking* that represents the opinion of the group in the best manner, with respect to how the items should be ordered. Rank aggregation is rooted in the voting theory field and early works of Jean-Charles de Borda (1733–1799) and Marie Jean Antoine Nicolas de Caritat (1743–1794), better known as the Marquis de Condorcet. These two politicians presented two very important ideas in voting theory. Borda's proposal was to determine winner of an election by assigning scores to the candidates based on what rank position they occupy in each ballot. Condorcet's proposal, known as the Condorcet criterion, was to determine the winner based on the majority's preferences over the candidate pairs. Formally, the Condorcet criterion declares that a candidate that is ranked ahead of every other candidate in the pairwise comparisons by a majority of votes must be declared as top-ranked candidate in the outcome of the election; such a candidate is denoted as the *Condorcet Winner*. Additionally, Condorcet presented a related important concept known as *Condorcet Paradox* upon observing that a majority of voters can prefer candidate x to candidate y and a majority of voters can prefer candidate y to candidate z, and still a majority of voters can prefer candidate

z to candidate x, causing a paradoxical outcome. A milestone in the field of social choice theory emerged years later in the works of Kenneth Arrow (Brandt *et al.*, 2016), especially the *Arrow's Impossibility Theorem* (Arrow, 1951). This theorem is motivated by the intuitive requirement that any reasonable *social welfare function* (SWF)—a function that maps the ranking ballots into one consensus total ordering of the candidates—should simultaneously be *weakly Paretian* and *independent of irrelevant alternatives (IIA)*. The weakly Paretian paradigm states that if all the voters strictly prefer candidate x to candidate y, then x must be strictly ranked ahead of y in the consensus ranking. The IIA paradigm states that the relative ordering of x and y in the consensus ranking only depends on the relative ordering of x and y in the input votes and nothing else; for example, not a third candidate z. Arrow (1951) proved that the only aggregation function that simultaneously is weakly Paretian and IIA is *dictatorship*, which is a voting rule if there exists a single fixed voter (i.e., dictator) whose vote is returned as the consensus vote regardless of other votes. The Impossibility Theorem demonstrated that certain basic and desirable properties of voting systems are incompatible. Simply put, there exists no perfect voting rule or aggregation function! Hence, many researchers have developed different social welfare functions that relax some of these desirable properties.

It is needless to say that the above concepts and fundamental theoretical results have important implications. Consider the case of voting rules in U.S. elections, which have garnered significant attention. Recently, the states of Alaska and Maine adopted a method called Ranked-Choice Voting (RCV) (Hare, 1861) for their primary and congressional elections (ALA, 2022; Mai, 2022), and more than twenty other states use RCV for their local elections (Ran, 2022). RCV is a multi-round voting rule where the candidate who receives the lowest number of first-places in each round is dropped from all ballots for the next round—the relative ordering of the remaining candidates

remains unchanged— until a surviving candidate is ranked first in the majority of the votes. While RCV fails to satisfy the Condorcet criterion and *monotonicity*—if candidate x is the winner of an election, changing a vote to rank x first never causes x to lose the election— it is nonetheless regarded as being superior to the plurality or first-past-the-post voting methods (Brandt *et al.*, 2016).

Over the past recent years, rank aggregation has found widespread applications in other fields outside of voting theory and computational social choice. It has been advocated as a systematic approach to guide decision-making processes, especially in multi-criteria decision-making (MCDM) (Benítez-Fernández and Ruiz, 2020; Chen *et al.*, 2013; Liao and Wu, 2020; Mohammadi and Rezaei, 2020). MCDM methods evaluate alternatives based on predefined criteria and subsequently sort or rank them based on the evaluations (Mohammadi and Rezaei, 2020). Prominent examples include AHP (analytic hierarchy process) (Saaty, 1977), ANP (analytic network process) (Saaty, 2001), and ELECTRE (ELimination and Choice Expressing REality) (Figueira *et al.*, 2016). Since different MCDM methods produce conflicting rankings, finding an overall consensu ranking that resolves these disagreements is of paramount importance (Mohammadi and Rezaei, 2020). In the context of information retrieval, analogous concerns fall under the umbrella of data fusion, where the goal is to derive a collective ranking of different information retrieval systems (Klementiev *et al.*, 2008); rank aggregation methods have been effective in this context as well (Hsu and Taksa, 2005). Furthermore, rank aggregation has also gained attention as a robust mechanism for consolidating heterogeneous ordered output lists by different machine learning techniques (Klementiev *et al.*, 2008). Related applications within this context include meta-search engines and spam detection (Desarkar *et al.*, 2016; Dwork *et al.*, 2001), feature selection (Bolón-Canedo and Alonso-Betanzos, 2019; Sarkar *et al.*, 2014; Onan and Korukoğlu, 2017), natural language processing

(Cascaro *et al.*, 2019; Mehta and Majumder, 2019), and label ranking (Aledo *et al.*, 2017; Werbin-Ofir *et al.*, 2019). Furthermore, rank aggregation has other applications such as crowdsourcing (Chatterjee *et al.*, 2018; Kemmer *et al.*, 2020), bioinformatics (Dimitrakopoulos *et al.*, 2018; Galdi *et al.*, 2019; Li *et al.*, 2019; Quillet *et al.*, 2020), journal rankings (Aledo *et al.*, 2018; Cook *et al.*, 2010), university rankings (Tavanaei *et al.*, 2018), supplier selection (Peng *et al.*, 2011), and network inference (Marbach *et al.*, 2012; Puerta *et al.*, 2021).

Rank aggregation methodologies are broken down into two categories: *distance-based* and *ad hoc* methods (Cook, 2006). The techniques in the latter category are further divided into elimination and non-elimination methods. A prominent example of elimination ad hoc methods is RCV; furthermore, the popular *score-based* methods fall into non-elimination ad hoc methods. Techniques in the score-based category rank items based on their scores according to a specified scoring function. Examples of score-based methods are the Borda rule (Borda, 1784) and the Copeland rule (Copeland, 1951). Techniques in the distance-based category aim to find a ranking among all possible rankings with the least distance to all input rankings according to a specified distance measure. There are various such measures between rankings including the Kemeny-Snell distance (Kemeny and Snell, 1962), the Kendall-tau distance (Kendall, 1938), and the Spearman's footrule distance(Diaconis and Graham, 1977). We refer the reader to Diaconis (1988) and Fagin *et al.* (2006) for more details on distance measures between rankings.

This work focuses on distance-based methods to take advantage of their rigorous axiomatic foundations and associated sociotheoretic properties (Brandt *et al.*, 2016) including being more robust to outliers and manipulation than ad hoc methods (Brandt *et al.*, 2016). An important aspect addressed in this dissertation is that distance-based aggregation problems tend to be more computationally demanding

and are often NP-hard (Brandt *et al.*, 2016).

Before proceeding, it is important to explain that rankings can be categorized as strict and non-strict. Strict rankings refer to the case where there are no ties, while non-strict rankings refer to the case where there *may* be ties. Both strict and non-strict rankings can even further be categorized as complete and incomplete, e.g. see Dwork *et al.* (2001), Moreno-Centeno and Escobedo (2016); all items are ranked in the former and some items *may* be unranked in the latter. Top-k lists are a popular form of incomplete rankings where only the ordering of the top-k ranked items are given—it is implicitly assumed that all items that are not in a top-k list are tied together for position $k+1$. Furthermore and as a convention, let a *full rank reversal* denote the case where two rankings fully disagree over the relative ordering of some distinct item-pair and a *partial rank reversal* denote the case where some distinct item-pair is tied in one ranking, but not in the other.

Kemeny and Snell (1962) introduced the first and perhaps the most popular distance-based rank aggregation framework. The authors proposed a set of axioms (non-negativity, triangular inequality, anonymity, extension, scaling, and commutativity) that must be satisfied by any distance metric on rankings, and introduced a distance that uniquely satisfies them. Rank aggregation using this distance metric is known as the Kemeny aggregation problem (KEMENY-AGG). The social welfare function represented by this problem uniquely satisfies five key social choice properties simultaneously: anonymity, neutrality, unanimity, reinforcement, and local stability (Brandt *et al.*, 2016; Young, 1988; Young and Levenglick, 1978). These theoretical benefits come at a high computational price as KEMENY-AGG is NP-hard even for only four input rankings (Bartholdi *et al.*, 1989; Dwork *et al.*, 2001). Due to this fact, many heuristic (e.g., see Aledo *et al.* (2019); Amodio *et al.* (2016); Ding *et al.* (2018a,b)) and approximation algorithms (e.g., see Ailon *et al.* (2008); Ailon (2010))

have been proposed. We refer the reader to Section 2.3.2 for a detailed review of existing approximation algorithms for KEMENY-AGG. It is worth adding that the Kendall-tau distance (Kendall, 1945) is another prominent distance measure, which is equivalent to the Kemeny-Snell distance in the space of strict rankings. Fagin *et al.* (2004) and Fagin *et al.* (2003) proposed two separate generalization of the Kendall-tau distance for comparing non-strict rankings and top-k lists, respectively. These two generalized distances and their associated rank aggregation problems serve as one of the backbones of this work.

There is also growing interest in exact solution techniques for KEMENY-AGG. In particular, KEMENY-AGG has been formulated as binary programming for strict rankings in Conitzer *et al.* (2006); Cook (2006); Pedings *et al.* (2012), and for non-strict rankings in Yoo and Escobedo (2021). Emond and Mason (2002) proposed a specialized branch and bound algorithm that implicitly enumerates all possible non-strict complete rankings. This method is only capable of handling small instances with up to 20 items; Yoo *et al.* (2020) modified this method for handling incomplete rankings. Other exact methods include the iterative algorithms of Azzini and Munda (2020) and Rico *et al.* (2022) for strict and non-strict rankings, respectively. It is worth adding that there are $n!$ and $0.5[(1.4)^{n+1}n!] >> n!$ strict and non-strict rankings. To highlight the increased difficulty of the rank aggregation problem engendered by this increased solution universe, when $n = 5$, there are 120 strict rankings and approximately 452 non-strict rankings; and when $n = 50$, there are 3.04×10^{64} strict rankings and approximately 4.31×10^{71} non-strict rankings.

A small number of works have focused on partitioning KEMENY-AGG based on certain social choice properties its optimal solution(s) is guaranteed to satisfy. Through these partitioning approaches, certain instances can be decomposed into a set of smaller subproblems while guaranteeing that solving them independently still

induces an optimal solution to the original problem. A notable scheme is based on the Extended Condorcet Criterion (XCC) proposed by Truchon (1998), who proved that the optimal solutions to KEMENY-AGG with strict rankings are consistent with XCC. Betzler *et al.* (2014) introduced another scheme based on the *3/4-Majority Rule* and concept of *dirty pairs*. The authors proved that the 3/4-Majority Rule cannot further partition an instance that has already been partitioned by XCC, meaning that XCC partitioning is always at least as good as partitioning using the 3/4-Majority Rule. Additionally, Milosz and Hamel (2020) introduced a related approach that finds the relative ordering of certain item-pairs in the optimal solution(s). While it was shown to be more effective than XCC in in providing the partial structure of the solution to KEMENY-AGG, its associated algorithm has a complexity of $O(n^3)$, whereas XCC has a complexity of $O(n^2)$—and it is only applicable for strict rankings. Recently, Yoo and Escobedo (2021) proposed the Non-strict Extended Condorcet Criterion (NXCC), which is a generalization of XCC for non-strict rankings, and introduced a partitioning algorithm based on sequential pairwise comparisons. The authors reported that whenever tested instances from the Preflib database (Mattei and Walsh, 2013) with up to 300 items were partitionable, the combined exact solution times of the decomposed subproblems—using their exact binary programming formulation—were at least 25% and up to 96% faster than those of the full problem.

Another important focus is obtaining high-quality lower bounds for the KEMENY-AGG, especially for assessing the solution quality of inexact methods. More specifically, a low-quality lower bound may lead to the incorrect conclusion that an optimal or near-optimal solution is of low quality. Davenport and Kalagnanam (2004) introduced the first lower bound using the pairwise comparison information. There are three general classes of lower bounding techniques for KEMENY-AGG: 1) pairwise comparison methods, 2) cycle-based methods, and 3) LP-relaxation methods. Pair-

wise comparison methods leverage the fact that each pair of alternatives contributes a minimum amount to the overall distance. Cycle-based methods seek to improve pairwise comparison lower bounding techniques by taking advantage of the fact that the preferences returned by the solution must be transitive. Finally, a lower bound can be obtained by solving the LP-relaxation of KEMENY-AGG formulation. We remark that there are other infrequently used lower bounding techniques with strict rankings such as using Spearman's footrule (Dwork *et al.*, 2001) and Borda count (Coppersmith *et al.*, 2010), based on their relationship with the Kemeny-Snell distance; however, to the best of our knowledge, these relationships have not been extended to the case of non-strict rankings.

In recent years, top-k lists have attracted significant attention due to their various advantages and real-world applications. Examples of top-k lists are the top-250 movies on IMDB or the top-10 played songs on Spotify (Pedroche and Conejero, 2020). Top-k lists have many advantages that can overcome some of the practical drawbacks of the traditional full-list approach: a collection of items may be too large to rank or even present, processing the full list could present a massive computational/cognitive load, and it may be impossible or meaningless to compare and rank items beyond a certain point (Chierichetti *et al.*, 2018). Due to the increased use of such lists, the top-k list aggregation problem (TOP-k-AGG) has attracted considerable attention. The aim of TOP-k-AGG is to find a *consensus list*, either another top-k list or a full list (i.e., an ordering of all n items), that best represents the input lists. TOP-k-AGG is interrelated with many problems in information retrieval including top-k recommendation (Chen *et al.*, 2022; Lee *et al.*, 2021; Kabra and Agarwal, 2021; Tang *et al.*, 2021) and top-k query (Mackenzie and Moffat, 2020; Shanbhag *et al.*, 2018; Xie *et al.*, 2020) problems. In top-k recommendation, a small number of items must be recommended among a larger set of available items to a user. Top-k recom-

mender systems are utilized by companies like Amazon, Netflix, and Hulu, and they are widely used in social networks and various other contexts (Song *et al.*, 2015). Recommender systems are hardly accurate due to data sparsity (Park *et al.*, 2016) and the results of multiple recommender systems are usually different from one another (Oliveira *et al.*, 2020); hence, there is growing interest in improving the quality of the recommendation list (e.g. see Ma *et al.* (2021); Zhu *et al.* (2019); Lee *et al.* (2021); Hu *et al.* (2018)). Recently, Oliveira *et al.* (2020) performed an extensive study on the effectiveness of rank aggregation techniques in top-k recommender systems. A total of 19 techniques were implemented and the quality of recommendations was improved on all but one of seven tested data sets. In top-k query, given a database and a query, the aim is to retrieve the top-k associated items in the database. Applications of top-k query processing include image retrieval (Zhang *et al.*, 2019), search engines (Long and Suel, 2003), and digital libraries (Lu and Callan, 2005). The relevance of the selected items to the given query is usually calculated by a scoring function. To neutralize the effect of using different algorithms or scoring functions, one can use TOP-k-AGG to consolidate these lists and derive a more coherent and effective list. TOP-k-AGG can be considered as an ensemble technique for those machine learning algorithms whose output is an ordered list.

1.2 Existing Challenges

Even though variations of the rank aggregation problem have been around for centuries, recent applications of this problem in various fields have prompted new practical challenges. One of the existing challenges is reliably solving large-scale instances of this problem mainly arising from the field of artificial intelligence. For example, there are various studies in bioinformatics to identify genes possibly associated with a certain disease. Since different studies identify different genes, there is

an essential need to combine the results of different studies instead of relying upon a single experiment, where each experiment may identify thousands of genes (Cohen-Boulakia *et al.*, 2011; Marbach *et al.*, 2012). Recommender systems (Oliveira *et al.*, 2020) and web query systems (Dwork *et al.*, 2001) are other fields where most rank aggregation instances have thousands of items. The exact methods of Kemeny-Agg are capable of solving instances mostly with tens and no more than a few hundred items reliably.. For example, the largest strict ranking instance solved exactly in Emond and Mason (2002), Conitzer *et al.* (2006), Betzler *et al.* (2014), had 15, 40, and 200 items, respectively, and the largest non-strict ranking instance solved exactly in Rico *et al.* (2022) and Yoo and Escobedo (2021) had 15 and 210 items, respectively. Condorcet partitions can be very useful for expediting exact techniques, particularly when the resulting partition has many small subsets. However, some instances are not partitionable, and in various other cases when they are, the partition may yield relatively few subsets and/or very large subsets. Yoo and Escobedo (2021) reported that a sizeable fraction of the real-world instances with ties drawn from the Preflib data set (Mattei and Walsh, 2013) yielded Condorcet partitions with these disadvantageous characteristics. Betzler *et al.* (2014) reported similar results on synthetic instances generated via the Plackett-Luce model (Luce, 2012; Plackett, 1975). Such results indicate that exact decomposition is useful only for a limited number of instances. Since exact methods fail to solve large-scale instances, even with the help of Condorcet partitions, one may reasonably turn to an approximation algorithm. However, the state-of-the-art approximation algorithm for Kemeny-Agg, LP-KwikSort (Ailon *et al.*, 2008), is only suitable for strict rankings and requires solving the LP-relaxed formulation, which again can only be used for instances with a few hundred items. On the other hand, the approximation ratio of other algorithms is not appealing: two for strict rankings and nine for non-strict rankings. To over-

come this challenge, this work proposes Approximate Condorcet Partitioning (ACP), a highly scalable decomposition-based solution technique with instance-specific solution guarantees that can reliably solve large-scale instances with thousands of items in seconds.

Another existing challenge comes in the form of handling ties as there is no unanimous agreement on how to handle partial rank reversals. In recent years, non-strict rankings have increased in prominence due to their enhanced flexibility for representing preference data. Dealing with non-strict rankings in real-world applications is the rule rather than the exception (D'Ambrosio *et al.*, 2019; Emond and Mason, 2002). In particular, it may not possible for humans to express their preferences strictly over more than a very small number of items, or a subset of items may be considered indistinguishable to a specific MCDM or machine learning algorithm (e.g, it may award the same score to multiple items). Moreover, forcing human judges to express their preferences in a strict manner may not reflect their true opinion. Therefore, developing rank aggregation frameworks capable of handling this type of ranking data is crucial. Setting aside the rather unacceptable process of breaking ties randomly, there are three prevalent treatments for handling partial rank reversal: 1) assuming full agreement (Kendall, 1942); 2) assuming complete disagreement (Andrieu *et al.*, 2021); and 3) Reflecting a level of agreement halfway between the two extremes (Kemeny and Snell, 1962). To elaborate, assume that every full rank reversal has unit weight. Then, each partial rank reversal has a weight of 0, 1, and 0.5 under treatments 1-3, respectively. Recently, Fagin *et al.* (2004) proposed the generalized Kendall-tau distance with parameter $0 \leq p \leq 1$, which contains the entire agreement-disagreement spectrum. Despite the high flexibility of this distance measure, its associated rank aggregation problem has received little to no attention in the literature. To overcome this gap in the literature, this work studies the rank aggregation problem under this

generalized distance for the first time, both theoretically and computationally.

TOP-k-AGG has been studied probabilistically (Chen *et al.*, 2019; Collas and Irurozki, 2021) and deterministically (Dwork *et al.*, 2001; Fagin *et al.*, 2003). In the probabilistic approach, it is assumed that the observed lists are realizations of a probabilistic model on ranking data, such as the Mallows model (Mallows, 1957), Thurstone (Thurstone, 1927), and Bradley-Terry-Luce (Bradley and Terry, 1952; Luce, 2012), and the goal is to recover the ground-truth list. Deterministic approaches are comprised of score-based and distance-based methods. Distance-based TOP-k-AGG techniques can be divided based on whether the output ranking is considered a full list or another top-k list. Dwork *et al.* (2001), Ailon (2010), and Nápoles *et al.* (2017) fall into the first category; Fagin *et al.* (2003) falls into the second category. Fagin *et al.* (2003)'s method provides higher flexibility, and it induces a far smaller solution space. There are $\binom{n}{k}k!$ possible top-k lists using the latter approach, which is $(n-k)!$ times smaller than $n!$ (the number of possible full strict lists over n). Despite the high flexibility of Fagin *et al.* (2003)'s method and its desirable applications in recommender systems—where the ordering of items at the bottom of the list is irrelevant— it has received little attention. This work facilitates the use of this method by proposing several exact and inexact solution techniques.

TOP-k-AGG has many real-world applications. A motivating example concerns competing university ranking lists, which are compiled by different services who apply various objective and subjective criteria and weights. Such lists can exhibit significant differences among one another and have strong implications on universities and students. Thus, it may be ill-advised to rely solely on one such source of information to judge the comparative standings of universities. To highlight this concern, consider a recent scandal that caused Columbia University to be dropped from rank 2 to 18 in the U.S. News Best National Universities list due to the submission of inaccurate data

by university officials (Col, 2022). The case was exposed by a whistleblower, suggesting that there might be other similar yet unreported instances since U.S. News and other services that publish university rankings rely on self-reported data.

TOP-k-AGG provides a mechanism for consolidating the information from multiple rankings of top universities. The version of the problem studied in this dissertation was first introduced in Fagin *et al.* (2003). It is worth elaborating on the sheer computational difficulty of this aggregation problem and the need for effective solution techniques for addressing it. The solution space is very large and difficult to explore; for example, one of the university instances solved herein has around 1.87×10^{198} possible solutions. Fagin *et al.* (2003) introduced a 2-approximation algorithm, which is to our knowledge the only known technique with rigorous guarantees for solving TOP-k-AGG. Applying it to the featured instance results in a solution that deviates by over 14% from the optimal solution—which is calculated using the exact methods introduced in Chapter 5 of this dissertation. To highlight the differences between the exact and approximate solutions to the 2022 university rankings TOP-k-AGG instance, four universities not in the top-10 in the optimal solution returned by the exact method appear in the suboptimal top-10 list list obtained via the approximate method. The top-ranked universities of the two methods differ as well. Harvard is the highest-ranked university in 2022 when using the exact method; however, Stanford University occupies the top position when using the approximate method.

1.3 Contributions and Overview of the Dissertation

This dissertation makes three main contributions to the study of rank aggregation problems. First, it derives a scalable solution approach for solving large-scale instances of KEMENY-AGG, while providing formal guarantees; its implications will enable the use of robust ranking aggregation to problems in artificial intelligence and

various other fields. Second, it facilitates the consideration of non-strict rankings by introducing and studying RANK-AGG, which can benefit a wide array of applications in MCDM. Third, it derives the first mathematical programming formulations and solution techniques for TOP-k-AGG, which may be used to mitigate the high variability of recommender and query processing systems. The ensuing paragraphs summarize the contents of the remaining four chapters.

Chapter 2 introduces the mathematical notation used throughout the dissertation. Furthermore, it provides an overview of distance-based and score-based rank aggregation frameworks. Furthermore, it reviews various prominent techniques for solving rank aggregation problems. Finally, it provides an overview of the social choice properties that KEMENY-AGG has been shown to satisfy. A few of these properties serve as one of the backbones of this dissertation.

Chapter 3 is devoted to deriving lower and upper bounding techniques for KEMENY-AGG. In particular, it introduces exact and approximate methodologies inspired by the social choice foundations of KEMENY-AGG; additionally, it generalizes existing lower bounds for strict rankings to the case of non-strict rankings, and it proposes shortcuts for reducing the run time of these techniques. This chapter makes several contributions: formalizing the concept of a finest-Condorcet partition and proposing an efficient algorithm for deriving it; developing Approximate Condorcet Partitioning (ACP), a scalable method for solving very large strict and non-strict rank aggregation instances; presenting formal solution guarantees (approximation factors) for any item-partitioning method; and providing improved solution guarantees for ACP. ACP is designed to solve very large-scale instances. Even though exact methods are only capable of handling a few hundred items, ACP is capable of producing high-quality solutions for instances with thousands of items[1]. As its final contribution, this chap-

[1]The contents of Sections 3.1-3.2 resulted in Akbari and Escobedo (2022a).

ter generalizes some of the existing lower bounding techniques for KEMENY-AGG with strict rankings for the case of non-strict rankings. Furthermore, it uses Condorcet criterion variations and a constraint relaxation method to accelerate the lower bounding process[2].

Chapter 4 delves into the rank aggregation problem under the generalized Kendall-tau distance (RANK-AGG). This chapter makes the following contributions: proposing an exact formulation of RANK-AGG; developing a new social choice property, which generalizes the Condorcet criterion and its variants beyond KEMENY-AGG; developing an algorithm for constructing a GXCC partition; and deriving new theoretical insights on the effect of the Kendall-tau distance penalty parameter[3].

Chapter 5 is dedicated to TOP-k-AGG. This chapter makes the following contributions: introducing a binary nonlinear programming formulation and four mixed-integer linear programming (MIP) formulations—two of which result from the introduction of a novel set of preference cycle-prevention constraints specific to TOP-k-AGG; comparing the strength of the MIP formulations using techniques from polyhedral theory; and introducing several heuristic algorithms and a data reduction technique for accelerating the solution to large-scale instances. Lastly, this work applies this framework to identify the top-100 U.S. universities by consolidating four prominent university rankings: U.S. News, Times Higher Education, QS World Ranking, and Academic Ranking of World Universities[4].

[2]The contents of Section 3.3 appeared in Akbari and Escobedo (2021).

[3]The contents of Chapter 4 resulted in Akbari and Escobedo (2022b).

[4]A shorter preliminary version of Chapter 5 led to Akbari and Escobedo (2022c).

Chapter 2

NOTATION AND PRELIMINARIES

This chapter is organized as follows. Section 2.1 describes basic mathematical notations to introduce the rank aggregation problems covered in the dissertation. Section 2.2 reviews the prominent distance-based rank aggregation methods. Section 2.3 describes the Kemeny-Snell distance and its associated aggregation problem, and Section 2.4 describes pertinent social choice-inspired decomposition schemes for these problems.

2.1 Mathematical Notation

Let $\boldsymbol{\mathcal{X}} = \{1, 2, \ldots, n\}$ be the set of items, $\boldsymbol{\mathcal{L}} = \{1, 2, \ldots, m\}$ be the set of indices of input rankings over $\boldsymbol{\mathcal{X}}$, and $\boldsymbol{\Sigma} \subset \mathbb{Z}^n$ be the set of all possible complete ranking vectors over $\boldsymbol{\mathcal{X}}$. Additionally, let $\boldsymbol{\sigma}^l$ be the input ranking $l \in \boldsymbol{\mathcal{L}}$, and σ_i^l be the rank of item i in $\boldsymbol{\sigma}^l$. As a convention, $i \succ_{\boldsymbol{\sigma}^l} j$ indicates that item i is preferred over item j in $\boldsymbol{\sigma}^l$, i.e., $\sigma_i^l < \sigma_j^l$, and $i \approx_{\boldsymbol{\sigma}^l} j$ indicates that i and j are tied in $\boldsymbol{\sigma}^l$, i.e., $\sigma_i^l = \sigma_j^l$. Furthermore, let $\boldsymbol{\Lambda} = \{(i, j) | i, j \in \boldsymbol{\mathcal{X}},\ j > i\}$ be the set of distinct item-pairs.

Definition 1 *Let $s_{ij} = |l \in L : i \succ_{\boldsymbol{\sigma}^l} j|$ and $t_{ij} = |l \in L : i \approx_{\boldsymbol{\sigma}^l} j|$ be the number of input rankings in which item i is preferred over item j, and the number of input rankings in which i and j are tied, respectively.*

Definition 2 *(Yoo and Escobedo, 2021) Item i is pairwise preferred by a decisive majority over item j if $s_{ij} > s_{ji} + t_{ij}$, that is, the number of input rankings which prefer i to j is greater than the number of input rankings which prefer j to i, plus*

those which tie them. If neither i is preferred over j nor j is preferred over i, then there is no decisive majority that prefers i over j, and vice versa.

For the rest of the dissertation, we use the term *pairwise preferred* instead of pairwise preferred by a decisive majority, for succinctness.

Definition 3 *Let $[c_{ij}] \in \mathbb{Z}^{n \times n}$ be the Cumulative Ranking (CR) matrix whose individual entries are obtained as $c_{ij} = s_{ij} + t_{ij} - s_{ji}$ when the input rankings are complete.*

The CR matrix is used to linearize KEMENY-AGG problem in Yoo and Escobedo (2021). Here, it is also employed to reduce the space requirements of one of the proposed algorithms (see Algorithms 1 and 2).

A sequence of subsets denoted as $\boldsymbol{X} = \{X_1, X_2, \ldots, X_w\}$ is a partition of $\boldsymbol{\mathcal{X}}$ if $\cup_{k=1}^{w} X_k = \boldsymbol{\mathcal{X}}$ and $X_k \cap X'_k = \emptyset, \forall k, k' \in \{1, \ldots, w\}$, with $k \neq k'$. Subset X_k is said to be preferred over subset X'_k, written as $X_k \succ X'_k$, if all items in X_k are pairwise preferred over all items in X'_k. Similar to Laslier (1997), we call partition $\boldsymbol{X}$ a *null* partition if $|\boldsymbol{X}| = 1$, a *trivial* partition if $|\boldsymbol{X}| = n$, and a *proper* partition otherwise.

2.2 Distance-Based Approaches

Definition 4 *(Distance-Based Rank Aggregation) Given $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{L}}$, let $d(.,.)$ be a distance measure between rankings. The distance-based rank aggregation problem seeks to find a complete ranking $\boldsymbol{\sigma}^* \in \boldsymbol{\Sigma}$ with the lowest cumulative distance to the input rankings; it can be written succinctly as*

$$\boldsymbol{\sigma}^* = \underset{\boldsymbol{\sigma} \in \boldsymbol{\Sigma}}{\operatorname{argmin}} \sum_{l \in \boldsymbol{\mathcal{L}}} d(\boldsymbol{\sigma}, \boldsymbol{\sigma}^l) \tag{2.1}$$

The rest of this section reviews three prominent distance measures between rankings, namely the Kemeny-Snell distance, Kendall-tau distance, and Spearman's footrule distance.

Definition 5 *The Kemeny-Snell distance (Kemeny and Snell, 1962) between two complete rankings* $\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2$*, denoted by* $d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$*, is given by*

$$d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \frac{1}{2} \sum_{i,j \in \boldsymbol{\mathcal{X}}} \left| sign(\sigma_i^1 - \sigma_j^1) - sign(\sigma_i^2 - \sigma_j^2) \right|. \tag{2.2}$$

The function $sign(v)$ returns 1 if $v > 0$, -1 if $v < 0$, and 0 otherwise. In the case of strict rankings, d_{KS} counts the number of full rank reversals; in the case of non-strict rankings, every full rank reversal has twice the weight of every partial rank reversal.

Definition 6 *The Kendall-tau distance (Kendall, 1938) between two complete strict rankings* $\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2$*, denoted by* $d_{KT}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$*, is given by*

$$d_{KT}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \sum_{(i,j) \in \boldsymbol{\Lambda}} K_{i,j}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2), \tag{2.3}$$

where $K_{i,j}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ *is set to 1 if the relative orderings of* i *and* j *are different in* $\boldsymbol{\sigma}^1$ *and* $\boldsymbol{\sigma}^2$*, and 0 otherwise. In other words, we have*

$$K_{i,j}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \begin{cases} 1 & (i \succ_{\boldsymbol{\sigma}^1} j \ \wedge \ j \succ_{\boldsymbol{\sigma}^2} i) \ \ \vee \ \ (j \succ_{\boldsymbol{\sigma}^1} i \ \wedge \ i \succ_{\boldsymbol{\sigma}^2} j) \\ 0 & \textit{otherwise.} \end{cases}$$

As stated earlier, whenever the input rankings are strict, d_{KT} and d_{KS} are equivalent; however, unlike d_{KS}, d_{KT} is not capable of handling ties (Fagin *et al.*, 2004). In two separate works, Fagin *et al.* (2003) and Fagin *et al.* (2004) proposed two generalization of the Kendall-tau distance for comparing top-k lists and non-strict rankings, respectively. These generalized distances and their associated rank aggregation problems are studied in Chapter 5 and Chapter 4, respectively.

Definition 7 *The Spearman's footrule distance between two strict rankings* $\boldsymbol{\sigma}^1$ *and*

$\boldsymbol{\sigma}^2$, *denoted by* $F(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$, *is defined as*

$$F(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \sum_{i \in \boldsymbol{\mathcal{X}}} |\sigma_i^1 - \sigma_i^2|,$$

that is the sum of absolute difference between the rank of all items in the two rankings.

Given two strict rankings $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$, we have (Diaconis and Graham, 1977)

$$d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) \leq F(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) \leq 2d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2).$$

It is worth mentioning that the distance-based rank aggregation problem has an interesting statistical interpretation. To that end, we first review the Mallows model (Mallows, 1957), which is a popular probabilistic model on ranking data and has the nice property of scalability (Marden, 2014). The model induces a distribution on $\boldsymbol{\Sigma}$ such that

$$Pr[\boldsymbol{\sigma} \in \boldsymbol{\Sigma}] = \frac{1}{Z_\beta} e^{-\beta . d(\boldsymbol{\sigma}', \boldsymbol{\sigma})}, \tag{2.4}$$

where $\beta > 0$ is the decay parameter, d is a distance measure between rankings, $\boldsymbol{\sigma}'$ is the ground-truth permutation, and Z_β is a normalizing constant. Mallows model specifies that the probability of observing a permutation has a reverse relationship with its distance from the ground-truth ranking. As β approaches zero, the distribution gets closer to a uniform distribution on $\boldsymbol{\Sigma}$, i.e., any permutation has equal probability of occurring, and as β approaches infinity, the distribution becomes more concentrated around $\boldsymbol{\sigma}'$ (Chierichetti *et al.*, 2014). Using distance d, the solution to the distance-based rank aggregation problem is equivalent to the maximum likelihood estimator of the Mallows probabilistic model, assuming the input rankings are

independent realizations (Braverman and Mossel, 2009), that is,

$$\boldsymbol{\sigma}^* = \underset{\boldsymbol{\sigma}\in\boldsymbol{\Sigma}}{\operatorname{argmin}} \sum_{l\in\boldsymbol{\mathcal{L}}} d(\boldsymbol{\sigma}, \boldsymbol{\sigma}^l) = \underset{\boldsymbol{\sigma}\in\boldsymbol{\Sigma}}{\operatorname{argmax}}\ \Pi_{l\in\boldsymbol{\mathcal{L}}} Pr[\boldsymbol{\sigma}^l]. \tag{2.5}$$

A systematic way of comparing the performance of ranking aggregation methods is to generate random instances by sampling a set of input rankings from a Mallows distribution (e.g., see D'Ambrosio *et al.* (2017); Yoo and Escobedo (2021)).

2.3 Kemeny Aggregation

KEMENY-AGG is a popular variant of the distance-based rank aggregation problem, which is capable of handling ties and satisfying various desirable properties (Brandt *et al.*, 2016). Section 2.3.1 introduces the components of this formal framework, and Section 2.3.2 reviews notable approximation algorithms, which will serve to introduce subsequent chapters.

2.3.1 Underlying Distance and Properties

KEMENY-AGG, i.e., rank aggregation under the Kemeny-Snell distance, has received significant attention as its solution uniquely satisfies five key social choice properties simultaneously: anonymity, neutrality, unanimity, reinforcement, and local stability. Moreover, Kemeny and Snell (1962) showed that the d_{KS} distance uniquely satisfies the following set of axioms:

Axiom 1: (Non-negativity) $d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) \geq 0$; and equality holds if and only if $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ are the same ranking.

Axiom 2: (Commutativity) $d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = d(\boldsymbol{\sigma}^2, \boldsymbol{\sigma}^1)$.

Axiom 3: (Triangular inequality) $d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) + d(\boldsymbol{\sigma}^2, \boldsymbol{\sigma}^3) \geq d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^3)$.

Axiom 4: (Anonymity) if $\boldsymbol{\sigma}^{1'}$ results from $\boldsymbol{\sigma}^1$ by a permutation of the objects in V,

and $\boldsymbol{\sigma}^{2'}$ results from $\boldsymbol{\sigma}^2$ by the same permutation, then $d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = d(\boldsymbol{\sigma}^{1'}, \boldsymbol{\sigma}^{2'})$.

Axiom 5: (Extensions) If two rankings $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ agree except from a set S of k elements, then $d(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ must be computed as if these k objects were the only objects being ranked.

Axiom 6: (Scaling) This minimum positive distance is 1.

Definition 8 *The optimal ranking obtained from* KEMENY-AGG *can be mathematically stated as*

$$\boldsymbol{\sigma}^*_{KEM} = \underset{\boldsymbol{\sigma} \in \boldsymbol{\Sigma}}{\operatorname{argmin}} \sum_{l \in \boldsymbol{\mathcal{L}}} d_{KS}(\boldsymbol{\sigma}, \boldsymbol{\sigma}^{\boldsymbol{l}}). \tag{2.6}$$

The solution to Problem (2.6) is known as the *Kemeny ranking*, *Kemeny consensus*, and *Kemeny median* in the literature.

Definition 9 *Let d^*_{KS} denote the Kemeny-Snell distance of $\boldsymbol{\sigma}^*_{KEM}$ to all of the input rankings.*

The d_{KS} distance has been shown to be related to the Kendall-tau correlation coefficient (τ_x) for complete rankings (Emond and Mason, 2002). The Kendall-tau correlation coefficient between two complete rankings $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ is defined as

$$\tau_x(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \sigma^1_{ij} \sigma^2_{ij}}{n(n-1)}, \tag{2.7}$$

where n is the number of items and σ_{ij} is defined as

$$\sigma_{ij} = \begin{cases} 1 & \text{if } \sigma_i \leq \sigma_i, \\ -1 & \text{if } \sigma_i > \sigma_i, \\ 0 & \text{if } i = j. \end{cases} \tag{2.8}$$

$\tau_x(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ and $d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ are related via the following equation (Emond and Mason, 2002):

$$\tau_x(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 1 - \frac{2d_{KS}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)}{n(n-1)}. \tag{2.9}$$

Leveraging Eq. (2.9), we have (Emond and Mason, 2002)

$$\underset{\boldsymbol{\sigma}}{\operatorname{argmin}} \sum_{l \in \boldsymbol{\mathcal{L}}} d_{KS}(\boldsymbol{\sigma}, \boldsymbol{\sigma}^l) = \underset{\boldsymbol{\sigma}}{\operatorname{argmax}} \sum_{l \in \boldsymbol{\mathcal{L}}} \tau_x(\boldsymbol{\sigma}, \boldsymbol{\sigma}^l). \tag{2.10}$$

Yoo and Escobedo (2021) exploited this relationship to propose the Generalized Kemeny Binary Programming (GKBP) formulation for solving KEMENY-AGG, which serves as the state-of-the-art formulation of KEMENY-AGG, to the best of our knowledge. Furthermore, Yoo *et al.* (2020) extended this relation to the case of incomplete rankings. GKBP is given by:

$$\max \quad z = \sum_{i,j \in \boldsymbol{\mathcal{X}}} c_{ij}(2y_{ij} - 1) \tag{2.11a}$$

$$\text{s.t. } y_{ij} - y_{kj} - y_{ik} \geq -1 \qquad \forall i, j, k \in \boldsymbol{\mathcal{X}}, \quad i \neq j \neq k, \tag{2.11b}$$

$$y_{ij} + y_{ji} \geq 1 \qquad \forall i, j, \in \boldsymbol{\mathcal{X}}, \quad j > i, \tag{2.11c}$$

$$y_{ii} = 0 \qquad \forall i \in \boldsymbol{\mathcal{X}}, \tag{2.11d}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall i, j \in \boldsymbol{\mathcal{X}}, \quad i \neq j. \tag{2.11e}$$

Here, decision variable y_{ij} is equal to 1 if alternative i is ranked ahead or tied with alternative j and 0 otherwise; i and j are tied in the ranking if $y_{ij} = y_{ji} = 1$. Objective function (2.11a) maximizes the extended Kendall tau correlation coefficient. Constraint (2.11b) prevents preference cycles; this means that if i is ranked ahead of j and j is ranked ahead of k, then i must be ranked ahead of k as well. Con-

straint (2.11c) enforces that for each pair of alternatives, one of them must be ranked ahead, after, or tied with the other. Constraint (2.11e) determines the domain of the variables. The rank of alternative i can be calculated as $\sigma_i = n - \sum_{j \neq i} y_{ij}$. To be the best of our knowledge, GKBP is the state-of-the-are formulation for solving KEMENY-AGG.

Definition 10 *Let $d_{KS}(\boldsymbol{\sigma})$ be the cumulative Kemeny-Snell distance of a given ranking $\boldsymbol{\sigma} \in \boldsymbol{\Sigma}$ to the input rankings; it is useful to also expand $d_{KS}(\boldsymbol{\sigma})$ as $\sum_{(i,j)\in\boldsymbol{\Lambda}} d_{KS}(\sigma_{ij})$, where $d_{KS}(\sigma_{ij})$ is the contribution of each pair of distinct item-pair $(i,j) \in \boldsymbol{\Lambda}$ in $d_{KS}(\boldsymbol{\sigma})$, which is given by*

$$d_{KS}(\sigma_{ij}) = \begin{cases} 2s_{ji} + t_{ij} & \text{if } \; i \succ_{\boldsymbol{\sigma}} j, \\ 2s_{ij} + t_{ij} & \text{if } \; j \succ_{\boldsymbol{\sigma}} i, \\ s_{ij} + s_{ji} & \text{if } \; i \approx_{\boldsymbol{\sigma}} j. \end{cases} \tag{2.12}$$

Eq. (2.12) follows from the definition of d_{KS} (see Eq. (2.2)). Intuitively, d_{KS} assigns a weight of 2 for each full rank reversal and a weight of 1 for each partial rank reversal. Therefore, if i is ranked ahead of j in $\boldsymbol{\sigma}$, the imposed distance for this pair equals the number of input rankings where j is ranked ahead of i, times 2, plus the number of input rankings where i and j are tied. Furthermore, if i and j are tied in $\boldsymbol{\sigma}$, the imposed distance for this pair equals the number of input rankings where either i is ranked ahead of j or vice versa.

2.3.2 Review of Notable Approximation Algorithms for KEMENY-AGG

Various methods with formal guarantees have been introduced to solve the rank aggregation problem induced by the Kemeny-Snell distance. First, we review approximation algorithms designed for KEMENY-AGG with strict rankings. The consensus

list obtained with the rank aggregation problem induced by the Spearman's footrule distance is a 2-approximation algorithm (Dwork *et al.*, 2001); this distance calculates the sum of the absolute differences between the rank positions assigned to each of the items. Ailon *et al.* (2008) proposed *KwikSort* and *LP-KwikSort*. KwikSort, an expected 2-approximation algorithm, repeatedly chooses a random item as the pivot, and it divides the remaining items into two groups—the sets of items ranked ahead and behind the pivot item, based on the pairwise comparison information; LP-KwikSort also chooses a random item as the pivot, and it divides the remaining items into two groups based on the linear programming (LP) relaxation solution of a KEMENY-AGG formulation. The authors proved that the best of KwikSort and *Pick-A-Perm* (see the next paragraph) yields an expected 11/7-approximation, and the best of LP-KwikSort and Pick-A-Perm yields an expected 4/3-approximation. In effect, KwikSort and LP-KwikSort do well on instances in which the Pick-A-Perm does not, and vice versa. Kenyon-Mathieu and Schudy (2007) derived the first polynomial time approximation scheme (PTAS) for the feedback arc set problem (FASP) on tournaments; a PTAS is a $(1 + \epsilon)$-approximation—i.e., it returns a solution up to $(1 + \epsilon)$ times the optimal objective function value, for any fixed value of $\epsilon > 0$. The authors also introduced a weighted generalization of the PTAS for FASP to provide the first PTAS for KEMENY-AGG. Because the time complexity of this PTAS is doubly exponential in $1/\epsilon$, its implementation becomes impractical for sufficiently small ϵ (Betzler *et al.*, 2014).

Next, we review approximation algorithms suitable for strict and non-strict rankings. Pick-A-Perm (Ailon *et al.*, 2008) selects one of the input rankings at random, and its deterministic version called *BestInput*, picks the input ranking with the lowest cumulative Kemeny-Snell distance to the input rankings. Pick-A-Perm and BestInput are expected 2-approximation algorithms for strict rankings; however, their ap-

proximation factors have not yet been defined for the case of non-strict rankings. Ailon (2010) proposed *RepeatChoice*, an expected 2-approximation algorithm, and *LPKwikSort*$_h$, an expected 3/2-approximation. While these two algorithms allow the input rankings to be non-strict, the consensus ranking is required to be strict, which is not be suitable for many applications. RepeatChoice repeatedly and without replacement chooses an input ranking and refines an initial non-strict ranking until all ties are broken. LPKwikSort$_h$ uses a novel LP rounding technique and is the first algorithm that, by itself, provides an (expected) approximation factor lower than 2. Van Zuylen and Williamson (2007) proposed a derandomized version of KwikSort, referred to herein as *DeterministicKwikSort*, which is an expected 2-approximation algorithm and showed that the best solution achieved by their algorithm and RepeatChoice provides an expected 8/5-approximation. Their work, similar to Ailon (2010), allows the input rankings to be non-strict but not the consensus ranking. Gionis *et al.* (2006) proposed BucketPivot, an expected 9-approximation algorithm, which is a generalization of KwikSort for non-strict rankings; its approximation factor reduces to 5 and 3 for special cases.

It is important to remark that, except for Spearman's footrule, the reviewed algorithms do not guarantee their respective approximation factor over all instances. Rather, their guarantees are achieved on average. We illustrate the potential for high variability in solution quality of such expected approximation algorithms using Pick-A-Perm. Let $\boldsymbol{\sigma}^1 = \cdots = \boldsymbol{\sigma}^9 = [1, 2, 3]^T$ and $\boldsymbol{\sigma}^{10} = [1, 3, 2]^T$. Here, $\boldsymbol{\sigma}^* = \boldsymbol{\sigma}^1$, with a cumulative Kemeny-Snell distance to the input rankings of 2. However, Pick-A-Perm may still choose $\boldsymbol{\sigma}^{10}$ (with a probability of 1/10), which has a cumulative Kemeny-Snell distance to the input rankings of 18 (9-times the expected factor).

2.4 The Condorcet Criterion and its Variants

The Condorcet criterion (CC), first proposed by Marquis de Condorcet (1785), is among the most prominent social choice properties. CC states that a candidate that is pairwise preferred over all other candidates must be declared as the top-ranked candidate, formally known as the *Condorcet Winner*. CC can be formally stated as (Young, 1988)

$$\text{if } \exists\, i \in \boldsymbol{\mathcal{X}} : \; s_{ij} > s_{ji} \;\; \forall j \in \boldsymbol{\mathcal{X}} \backslash \{i\} \implies i \succ_{\boldsymbol{\sigma}} j \;\; \forall j \in \boldsymbol{\mathcal{X}} \backslash \{i\},$$

where $\boldsymbol{\sigma}$ is the optimal ranking(s). A voting rule is said to be *Condorcet consistent* if it always selects the Condorcet Winner as the top-ranked item in its consensus ranking $\boldsymbol{\sigma}$, when one exists (Brandt *et al.*, 2016). Apart from KEM-AGG, there are other Condorcet consistent rank aggregation methods such as Dodgson's rule (Dodgson, 1876), maximin rule (Young, 1977), and the ranked pairs rule (Tideman, 2017).

Truchon (1998) proposed the Extended Condorcet criterion (XCC), which generalizes CC to guarantee an ordering of item-subsets in the aggregate ranking(s). XCC states that if $\boldsymbol{\mathcal{X}}$ can be arranged into a partition such that $X_k \succ X'_k$, $\forall k, k' \in \{1, \ldots, w\}$, with $k < k'$, then all items in X_k must be ranked ahead of all items in X'_k in the consensus ranking. XCC can be stated formally as:

$$\text{if } s_{ij} > s_{ji} \;\; \forall i \in X_k \; \forall j \in X'_k \; \forall k < k' \implies i \succ_{\boldsymbol{\sigma}} j \;\; \forall i \in X_k \; \forall j \in X'_k \; \forall k < k'.$$

Truchon (1998) proved that the solution to KEMENY-AGG satisfies XCC. Note that the exact ordering of the full set of items is determined by solving the separate KEMENY-AGG subproblems induced by the items in each subset of the partition.

Recently, Yoo and Escobedo (2021) showed that KEMENY-AGG for non-strict

rankings is inconsistent with XCC. That is, solutions to this problem, which allows rankings with and without ties, may violate XCC. The authors defined a social choice property called the Non-strict Extended Condorcet Criterion (NXCC), which can be stated formally as:

$$\text{if } s_{ij} > s_{ji} + t_{ij} \quad \forall i \in X_k \; \forall j \in X'_k \; \forall k < k' \implies i \succ_{\boldsymbol{\sigma}} j \quad \forall i \in X_k \; \forall j \in X'_k \; \forall k < k'.$$

Observe that, for the case with all strict rankings (i.e., $t_{ij} = 0 \; \forall i, j \in X$), NXCC becomes XCC. It was formally demonstrated in Yoo and Escobedo (2021) that the consensus rankings returned by KEMENY-AGG for non-strict rankings are consistent with NXCC.

Chapter 3

LOWER AND UPPER BOUNDING TECHNIQUES FOR KEMENY-AGG

This chapter focuses on lower and upper bounding techniques for KEMENY-AGG, and it makes four main related contributions. First, it improves Condorcet partitioning by defining the finest possible partition that is consistent with XCC and NXCC. This *finest-Condorcet partition* yields the most subsets among all such possible decompositions, maximizing their potential computational benefits to exact KEMENY-AGG approaches. Second, this chapter derives an efficient algorithm to construct the finest-Condorcet partition, which provides other structurally useful information. As its third main contribution, this chapter leverages these insights to introduce *Approximate Condorcet Partitioning (ACP)*, an efficient technique that can further decompose KEMENY-AGG instances whose finest-Condorcet partition contains one or more subsets that are too large to solve using exact methods. This contribution is accompanied by the derivation of instance-specific approximation factors, which are applicable to any item-partitioning scheme, including those that may not be consistent with Condorcet extensions. Improved guarantees are derived for the ACP solution; although these approximation factors are also instance-specific, their values were lower than those offered by all constant-factor approximation algorithms known to date, for all benchmark instances tested in this chapter. As a result, ACP serves a new upper bounding technique for KEMENY-AGG. As its fourth and final contribution, this chapter generalizes existing lower bounding techniques for KEMENY-AGG with strict rankings to non-strict rankings.

The rest of this chapter is organized as follows. Section 3.1 introduces the finest-Condorcet partition and proves its uniqueness; furthermore, it develops an efficient

algorithm to construct it. Section 3.2 introduces the Approximate Condorcet Partitioning technique and derives its provable guarantees, and compares this technique with various prominent approximation algorithms for KEMENY-AGG. Section 3.3 reviews some of the existing lower bounding techniques for KEMENY-AGG and extends these techniques to the case of non-strict rankings. Finally, Section 3.4 concludes the chapter.

3.1 The finest-Condorcet partition

Henceforth, we will denote partitions based on XCC and NXCC simply as *Condorcet Partitions* to distinguish them from those based on item exact and heuristic partitioning schemes (e.g. see Betzler *et al.* (2014) and Aledo *et al.* (2021)). The rest of this section is organized as follows. Section 3.1.1 formally introduces the concept of the finest-Condorcet partition, specifies its conditions, and proves that it is unique. Section 3.1.2 proposes a novel algorithm for obtaining the finest-Condorcet partition.

3.1.1 Definition and Properties

Let $\wp(\boldsymbol{\mathcal{X}})$ denote the class of partitions satisfying NXCC. This class can contain more than one partition; however, certain members of $\wp(\boldsymbol{\mathcal{X}})$ are more computationally expedient than others. In particular, after obtaining a Condorcet Partition, it is necessary to solve a KEMENY-AGG subproblem for each subset of the partition and then to concatenate the separate solutions, in proper order, to obtain a solution to the original problem. The worst case happens when the instance has a null Condorcet Partition ($|\mathbf{X}| = 1$), and the best case happens when the instance has a trivial Condorcet Partition ($|\mathbf{X}| = n$) (i.e., the order of the singleton subsets in the partition provides the optimal ranking of all items). For this reason, it is desirable to obtain partitions with more subsets and/or with smaller subsets. The ensuing example il-

lustrates the differences between multiple NXCC partitions and motivates our focus on the finest among all such partitions.

Example 1 *Consider an instance with 6 rankings of 6 items. The input rankings and the pairwise comparison matrices,* $\boldsymbol{S} = [s_{ij}] \in \mathbb{Z}^{6\times 6}$ *and* $\boldsymbol{T} = [t_{ij}] \in \mathbb{Z}^{6\times 6}$*, are given by*

Item	Input Rankings					
	$\boldsymbol{\sigma}^1$	$\boldsymbol{\sigma}^2$	$\boldsymbol{\sigma}^3$	$\boldsymbol{\sigma}^4$	$\boldsymbol{\sigma}^5$	$\boldsymbol{\sigma}^6$
1	1	1	1	1	1	1
2	3	3	3	2	2	1
3	2	2	4	4	2	2
4	3	3	3	3	3	5
5	4	4	2	4	5	3
6	5	5	5	5	4	4

$$\mathbf{S} = \begin{bmatrix} 0 & 6 & 6 & 6 & 6 & 6 \\ 0 & 0 & 3 & 3 & 5 & 6 \\ 0 & 2 & 0 & 4 & 4 & 6 \\ 0 & 0 & 2 & 0 & 4 & 5 \\ 0 & 1 & 1 & 2 & 0 & 5 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}, \mathbf{T} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 3 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

There are seven NXCC partitions: $\boldsymbol{X}^1 = \{\{1\},\{2,3,4\},\{5\},\{6\}\}$*,* $\boldsymbol{X}^2 = \{\{1,2,3,4\},\{5\},\{6\}\}$*,* $\boldsymbol{X}^3 = \{\{1,2,3,4,5\},\{6\}\}$*,* $\boldsymbol{X}^4 = \{\{1\},\{2,3,4\},\{5,6\}\}$*,* $\boldsymbol{X}^5 = \{\{1,2,3,4\},\{5,6\}\}$*,* $\boldsymbol{X}^6 = \{\{1\},\{2,3,4,5\},\{6\}\}$*, and* $\boldsymbol{X}^7 = \{\{1,2,3,4,5,6\}\}$*. The first is the finest and most desirable, as it has the highest number of subsets; indeed, the only subproblem that needs to be solved is the one corresponding to items 2, 3, and 4 (since the other subsets are singletons). Notice that it is possible to further partition at least one subset in* $\boldsymbol{X}^2$*-*$\boldsymbol{X}^7$ *while satisfying NXCC.*

The concept of a *finest-Condorcet partition* was first introduced in Truchon (1998), although its formal definition or required conditions were not provided therein. This partition is an adaption of the unique *minimal decomposition* (Laslier, 1997) of a tournament, which is applicable only for aggregating strict rankings. Next, we formally define an extension of the finest-Condorcet partition that is suitable for both strict and non-strict rankings, and we specify its required conditions.

Definition 11 *Partition* $\boldsymbol{X}^f \in \wp(\boldsymbol{\mathcal{X}})$ *is the finest-Condorcet partition if there is no other partition* $\boldsymbol{X} \in \wp(\boldsymbol{\mathcal{X}})$ *such that* $|\boldsymbol{X}| > |\boldsymbol{X}^f|$*, that is,* $\boldsymbol{X}^f$ *is the partition with the*

most subsets.

For any $\mathbf{X} \in \wp(\boldsymbol{\mathcal{X}}) \backslash \mathbf{X}^f$—i.e., all but the finest partition of the class—it is possible to further decompose at least one of the subsets such that the resulting partition still satisfies NXCC. To improve both XCC and NXCC, we add a requirement that is only satisfied by $\mathbf{X}^f$. Let $\wp^f(\boldsymbol{\mathcal{X}})$ be the class of finest-Condorcet partitions. Any $\mathbf{X}^f \in \wp^f(\boldsymbol{\mathcal{X}})$ must satisfy

$$\forall X_k \in \mathbf{X}^f,\ \nexists \overline{X}_k \subset X_k :\ s_{ij} > s_{ji} + t_{ij}, \forall i \in \overline{X}_k, \forall j \in X_k \backslash \overline{X}_k. \tag{3.1}$$

Condition (3.1) does not allow a subset of items in X_k to be pairwise preferred over the rest of the items in X_k, for all $X_k \in \mathbf{X}^f$ (i.e., the subsets cannot be further decomposed while satisfying NXCC). Later we prove that NXCC and Condition (3.1) are the necessary and sufficient conditions for the finest-Condorcet partition. Beforehand, Theorem 1 proves that $|\wp^f(\boldsymbol{\mathcal{X}})| = 1$, meaning that $\mathbf{X}^f$ is unique.

Theorem 1 *The finest-Condorcet partition is unique. Let* $\boldsymbol{X}^f, \boldsymbol{X}' \in \wp^f(\boldsymbol{\mathcal{X}})$*, where* $\boldsymbol{X}^f = \{X_1^f, X_2^f, \ldots, X_w^f\}$, $\boldsymbol{X}' = \{X_1', X_2', \ldots, X_w'\}$, *and* $\boldsymbol{X}^f \neq \boldsymbol{X}'$. *Since both* $\boldsymbol{X}^f$ *and* $\boldsymbol{X}'$ *are distinct finest-Condorcet partitions, they must have the same number of subsets, but the contents of some of their subsets must be different.*

Consider X_1^f *and* X_1'. *If* $X_1^f = X_1'$, *this part of the proof is trivially satisfied. Otherwise, assume that* $X_1^f \neq X_1'$ *and consider two cases based on the relative cardinality of the subsets.*

Case 1. $|X_1^f| = |X_1'|$. *There exist items* i *and* j *such that* $i \in X_1^f, i \notin X_1'$ *and* $j \notin X_1^f, j \in X_1'$. *This implies that* i *is pairwise preferred over* j *and* j *is pairwise preferred over* i*, a contradiction.*

Case 2. $|X_1^f| \neq |X_1'|$. *Without loss of generality assume that* $|X_1^f| > |X_1'|$.

Case 2.1: $X_1' \subset X_1^f$. *In this case, the contents of the respective partitions are given*

by

$$\boldsymbol{X}^f = \{\overbrace{\{X'_1,\, X_1^f \backslash X'_1\}}^{X_1^f}, \ldots, X_w^f\},$$
$$\boldsymbol{X}' = \{X'_1, \ldots, \underbrace{\{X_1^f \backslash X'_1, X'_k \backslash (X_1^f \backslash X'_1)\}}_{X'_k}, \ldots, X'_w\}.$$

Without loss of generality, assume that $X_1^f \backslash X'_1 \subset X_{k'}$ (considering a subset of $X_1^f \backslash X'_1$ also works). The $\boldsymbol{X}'$ is a finest-Condorcet partition and, therefore, all items in X'_1 are pairwise preferred over all items in $X_1^f \backslash X'_1$; thus, it is possible to decompose X_1^f and obtain a finer partition, contradicting the assumption that $\boldsymbol{X}^f$ is a finest-Condorcet partition.

Case 2.2: *$X'_1 \not\subset X_1^f$. This leads to a similar contradiction as in Case 1, since there exists items i and j such that $i \in X_1^f, i \notin X'_1$ and $j \notin X_1^f, j \in X'_1$.*

These cases prove that $X_1^f = X'_1$. Next, consider partitions $\boldsymbol{X}^f \backslash X_1^f$ and $\boldsymbol{X}' \backslash X'_1$ and apply the above chain of arguments to show that $X_2^f = X'_2$. Continuing in this manner gives that $X_k^f = X'_k$, for $k = 3, \ldots, w$. Therefore, we can conclude that the finest-Condorcet partition is unique.

Theorem 2 *$\boldsymbol{X}^f \in \wp(\boldsymbol{\mathcal{X}})$ is the finest-Condorcet partition if and only if it satisfies Condition (3.1).*

$\Longrightarrow$ We need to prove that if $\mathbf{X}^f$ satisfies Condition (3.1), then it is the finest-Condorcet partition. Note that $\mathbf{X}$ has the most subsets among all partitions in $\wp(\boldsymbol{\mathcal{X}})$ if it is not possible to further decompose its subsets. This is indeed equivalent to satisfying Condition (3.1).

$\Longleftarrow$ We need to prove that the finest-Condorcet partition must satisfy Condition (3.1). We use contradiction. Assume that at least one of the subset of $\mathbf{X}$, say X_k,

does not satisfy Condition (3.1). Then, we can further decompose X_k into $\overline{X}_k$ and $X_k \backslash \overline{X}_k$ and increase the size of $\mathbf{X}$ by 1. However, this contradicts the fact that $\mathbf{X}$ is the finest-Condorcet partition, as we can construct another valid partition that has more subsets.

3.1.2 An Efficient Algorithm for Constructing $\boldsymbol{X}^f$

This section presents an algorithm for constructing the finest-Condorcet partition. Beforehand, it is expedient to link the pairwise preference relationships, i.e., $s_{ij} > s_{ji} + t_{ij}$, with the elements of the CR matrix (see Subsection 2.1), namely to reduce storage requirements and computational effort.

Proposition 1 *Item i is pairwise preferred over item j if and only if $c_{ij} > 0$ and $c_{ji} < 0$.*

Recall that $s_{ij}, s_{ji}, t_{ij} \geq 0$, $t_{ij} = t_{ji}$, and $c_{ij} = s_{ij} + t_{ij} - s_{ji}$ $\forall i, j \in \boldsymbol{\mathcal{X}}$.

$\Longrightarrow$ Assume that $s_{ij} > t_{ij} + s_{ji}$.

Case 1. $t_{ij} = 0$. By substituting $s_{ij} > s_{ji}$ in the expressions for c_{ij} and c_{ji} we have that $c_{ij} > 0$, $c_{ji} < 0$.

Case 2. $t_{ij} > 0$. By substituting $s_{ij} > s_{ji} + t_{ij}$ in the expressions for c_{ij} and c_{ji} we have:

$$c_{ij} = s_{ij} + t_{ij} - s_{ji} > s_{ji} + t_{ij} + t_{ij} - s_{ji} = 2t_{ij} > 0,$$
$$c_{ji} = s_{ji} + t_{ij} - s_{ij} < s_{ji} + t_{ij} - s_{ji} - t_{ij} = s_{ji} - s_{ji} < 0$$

$\Longleftarrow$ Now, assume that $c_{ij} > 0$, $c_{ji} < 0$. Here, we have $c_{ij} = s_{ij} + t_{ij} - s_{ji} > 0$, which results in $s_{ij} > s_{ji} - t_{ij}$. Similarly, $c_{ji} = s_{ji} + t_{ij} - s_{ij} < 0$ results in $s_{ij} > s_{ji} + t_{ij}$. Since $s_{ji} + t_{ij} \geq s_{ji} - t_{ij}$, we can conclude that $s_{ij} > s_{ji} + t_{ij}$.

From Proposition 1, the CR matrix contains sufficient information to determine the pairwise preferences of all item-pairs and thereby enable Condorcet Partitioning. Its use reduces storage requirements since instead of having to store $[s_{ij}] \in \mathbb{Z}^{n\times n}$ and $[t_{ij}] \in \mathbb{Z}^{n\times n}$, only $[c_{ij}] \in \mathbb{Z}^{n\times n}$ is needed. Next, we define the parameters needed by the presented algorithm.

Definition 12 *Let Γ_i be the set of items over which item i is pairwise preferred; its contents are given by $\Gamma_i := \{j \in \boldsymbol{\mathcal{X}} : s_{ij} > s_{ji} + t_{ij}\}$, or equivalently, $\Gamma_i := \{j \in \boldsymbol{\mathcal{X}} : c_{ij} > 0,\, c_{ji} < 0\}$. Additionally, let $\gamma_i := |\Gamma_i|$ denote the number of items over which i is pairwise preferred.*

Definition 13 *Let $\overline{\Gamma}_i := \boldsymbol{\mathcal{X}} \backslash (\Gamma_i \cup \{i\})$ be the set of items over which item i is not preferred.*

The following proposition serves as the foundation of the proposed algorithm, which connects the γ-values of a distinct item-pair to their relative positions in the subsets of $\mathbf{X}^f$.

Proposition 2 *If $\gamma_i > \gamma_j$, then item j cannot belong to a lower-indexed subset than item i in $\boldsymbol{X}^f$; additionally, if $\gamma_i = \gamma_j$, then i and j must belong to the same subset.*

We prove this by contradiction. Let $\mathbf{X}^f = \{X_1, X_2, \ldots, X_w\}$ be the finest-Condorcet partition. Additionally, let item i to belong to $X_k \in \mathbf{X}^f$ and item j to belong to $X_{k'} \in \mathbf{X}^f$, where $k, k' \in \{1, \ldots, w\}$, with $k < k'$. This gives that

$$\mathbf{X}^f = \{X_1, \ldots, \underbrace{\{j, \ldots\}}_{X_k}, \ldots, \underbrace{\{i, \ldots\}}_{X_{k'}}, \ldots, X_w\}.$$

Letting $|X_k|$ be the number of items in subset X_k, bounds on γ_i and γ_j can be obtained

as

$$|X_{k+1}| + \cdots + |X_{k'}| + \cdots + |X_w| \leq \gamma_j \leq |X_k| + \cdots + |X_{k'}| + \cdots + |X_w| - 1, \text{ and}$$
$$|X_{k'+1}| + \cdots + |X_w| \leq \gamma_i \leq |X_{k'}| + \cdots + |X_w| - 1.$$

The lower bound on γ_j comes from the definition of $\mathbf{X}^f$, since each item in X_k must be pairwise preferred over all items in subsets $X_{k+1}, \ldots, X_w$. The upper bound on γ_j comes from the fact that, when $|X_k| > 1$, j can be pairwise preferred over some items in X_k, but there must be at least one item in this subset over which j is not pairwise preferred; otherwise, j must belong to X_{k-1}. Lower and upper bounds on γ_i are calculated in the same fashion. The values of γ_i and γ_j can be connected as follows:

$$\gamma_j \geq \sum_{t=k+1}^{w} |X_t| = \sum_{t=k+1}^{k'-1} |X_t| + \sum_{t=k'}^{w} |X_t| \geq \sum_{t=k'}^{w} |X_t| > \sum_{t=k'}^{w} |X_t| - 1 \geq \gamma_i.$$

Therefore, this gives that $\gamma_j > \gamma_i$, which contradicts the starting assumption. In summary, when $\gamma_i > \gamma_j$, j cannot belong to a lower-indexed subset than i in $\mathbf{X}^f$. Through a parallel chain of arguments, a similar contradiction results when $\gamma_i = \gamma_j$, meaning that i and j must belong to the same subset in the latter case. The pseudocode of the proposed partitioning procedure is presented in Algorithm 1, and it consists of two phases: 1) construction of an initial partition, 2) validation & merging. The algorithm utilizes Proposition 2 to build an initial partition $\mathbf{X}^0$. According to this proposition, if $\gamma_i = \gamma_j$, then i and j must belong to the same subset in $\mathbf{X}^f$; additionally, if $\gamma_i > \gamma_j$, j cannot belong to a lower-indexed subset than i in $\mathbf{X}^f$. Hence, $\{X_1^0, X_2^0, \ldots, X_w^0\} = \mathbf{X}^0$ is constructed by ordering the items by non-increasing γ-values; items with a distinct value are placed in separate subsets, and items with the same value are placed in the same subset. In more detail, the item(s) with the

Algorithm 1: Finest-Condorcet Partition

Input : $[c_{ij}] \in \mathbb{Z}^{n\times n}$ (CR matrix)
Output: Finest-Condorcet partition ($\mathbf{X}^f$), initial partition ($\mathbf{X}^0$)

Apply Definitions 12 and 13 to calculate parameters Γ_i, $\overline{\Gamma}_i$, and γ_i using $[c_{ij}]$, for $i \in \boldsymbol{\mathcal{X}}$;
Construct the initial partition $\mathbf{X}^0$ by placing all item(s) with the highest γ-value in X_1^0, all item(s) with the next highest γ-value in X_2^0, etc.;
$\widetilde{\mathbf{X}} \leftarrow \{X_1^0, X_2^0, \ldots, X_w^0\} = \mathbf{X}^0$; `// set working partition to initial partition`
$k = 1$;
while $k < |\boldsymbol{X}^0|$ **do**
  $\overline{\Gamma}(\widetilde{X}_k) \leftarrow \cup_{i\in\widetilde{X}_k}\overline{\Gamma}_i$; `// get items over which items in` $\widetilde{X}_k$ `are not pairwise preferred`
  if $\overline{\Gamma}(\widetilde{X}_k)\backslash \cup_{t=1}^{k} \widetilde{X}_t = \emptyset$; `// if` $\widetilde{X}_k$ `does not violate NXCC` **then**
    $k \leftarrow k + 1$;
  else
    while $\overline{\Gamma}(\widetilde{X}_k)\backslash \cup_{t=1}^{k} \widetilde{X}_t \neq \emptyset$ **do** ; `// while subset` $\widetilde{X}_k$ `violates NXCC`
      $k' \leftarrow \max\{g : i \in \overline{\Gamma}(\widetilde{X}_k)\backslash \cup_{t=1}^{k} \widetilde{X}_t \wedge i \in \widetilde{X}_g\}$; `// get max-index subset where a violation is detected`
      $\widetilde{X}_k \leftarrow \cup_{t=k}^{k'}\widetilde{X}_t$; `// merge subsets causing NXCC violation into` $\widetilde{X}_k$
    $k \leftarrow k' + 1$;
$\mathbf{X}^f \leftarrow \widetilde{\mathbf{X}}$;
return $\boldsymbol{X}^f, \boldsymbol{X}^0$

maximum γ-value are placed in X_1^0, item(s) with the next highest value is placed in X_2^0, etc. The second phase checks whether $\mathbf{X}^0$ satisfies NXCC; if it does not, it merges the subsets that have caused the violation. This process is repeated until the working partition satisfies NXCC. The next two theorems prove that Algorithm 1 is correct, meaning that its output satisfies NXCC and Condition (3.1) (even though this algorithm never checks Condition (3.1) in its process), and it has a time complexity of $O(n^2)$.

Theorem 3 *Algorithm 1 is correct.*

Assume that the initial partition $\mathbf{X}^0 = \{X_1^0, X_2^0, \ldots, X_w^0\}$ has been accordingly constructed, per Algorithm 1, and let $\widetilde{\mathbf{X}}$ be the working partition which is initially set to $\mathbf{X}^0$. Let $\overline{\Gamma}(\widetilde{X}_k) = \cup_{i \in \widetilde{X}_k} \overline{\Gamma}_i$ denote the set of items over which at least one of the items in $\widetilde{X}_k \in \widetilde{\mathbf{X}}$ is not pairwise preferred. If all items in $\overline{\Gamma}(\widetilde{X}_k)$ belong to $\widetilde{X}_k$ or to other lower-indexed subsets, $\widetilde{\mathbf{X}}$ does not violate NXCC, otherwise, there is a violation. Whenever a violation is detected, the associated subsets (see the next paragraph) are merged. The process continues until the working partition does not violate NXCC.

The validation & merging starts from subset $\widetilde{X}_1$. Generally, if $\overline{\Gamma}(\widetilde{X}_k) \backslash \cup_{t=1}^{k} \widetilde{X}_t = \emptyset$, all items in $\widetilde{X}_k$ are pairwise preferred over all items in $\widetilde{X}_t$, for $t = k{+}1, \ldots, |\widetilde{\mathbf{X}}|$. Therefore, subset $\widetilde{X}_k$ satisfies NXCC and remains unchanged in this case. If $\overline{\Gamma}(\widetilde{X}_k) \backslash \cup_{t=1}^{k} \widetilde{X}_t \neq \emptyset$, there is at least one item in subsets $\widetilde{X}_{k+1}, \ldots, X_{|\widetilde{\mathbf{X}}|}$ that not all items in $\widetilde{X}_k$ are pairwise preferred over, which causes a violation of NXCC. Let $\widetilde{X}_{k'}$ be the highest-indexed subset to which an item from $\overline{\Gamma}(\widetilde{X}_k) \backslash \cup_{t=1}^{k} \widetilde{X}_t$ belongs, where $k' > k$. Therefore, subsets $\widetilde{X}_k, \ldots, \widetilde{X}_{k'}$ are merged and placed into subset $\widetilde{X}_k$. Validation & merging is repeated for $\widetilde{X}_k$, which is now defined as $\cup_{t=k}^{k'} \widetilde{X}_t$, until $\overline{\Gamma}(\widetilde{X}_k) \backslash \cup_{t=1}^{k} \widetilde{X}_k = \emptyset$; this process is performed on the remaining subsets until the working partition satisfies NXCC. Hence, the output of the algorithm satisfies NXCC.

Furthermore, we prove that the output of the algorithm satisfies Condition (3.1). Recall that Condition (3.1) states that a subset cannot be further split into two subsets while satisfying NXCC. For this part of the proof, we emphasize that $\{X_1^f, X_2^f, \ldots, X_{|\mathbf{X}^f|}^f\} = \mathbf{X}^f$ refers to the partition output by the algorithm. Assume that subset $X_k^f \in \mathbf{X}^f$ has not undergone validation & merging, therefore, it remains unchanged after implementing the algorithm. If $|X_k^f| = 1$, then the subset trivially satisfies Condition (3.1). We use contradiction for the case when $|X_k^f| > 1$. Assume that subset X_k^f does not satisfy Condition (3.1), meaning it can be decomposed into

$\overline{X}_k^f$ and $X_k^f \backslash \overline{X}_k^f$. Consider items $i \in \overline{X}_k^f$ and $j \in X_k^f \backslash \overline{X}_k^f$. This gives that $\gamma_i > \gamma_j$, contradicting the fact that $\gamma_i = \gamma_j$ (since all items with the same γ-values were placed in the same subset in $\mathbf{X}^0$ and hence $\widetilde{\mathbf{X}}$).

Now, assume that subset X_k^f has undergone the merging process and that a sequence of consecutive subsets $\{\widetilde{X}_k, \ldots, \widetilde{X}_{k'}\} \subseteq \widetilde{\mathbf{X}}$ were merged to form subset X_k^f. Additionally, assume that X_k^f does not satisfy Condition (3.1) and, hence, it is possible to further decompose X_k^f into $\{X_k^f, \ldots, X_t^f\}$ and $\{X_{t+1}^f, \ldots, X_{k'}^f\}$ while satisfying NXCC. This contradicts the fact that X_k^f has triggered the merging process as at least one item in X_k^f is not pairwise preferred over at least one item in $X_{k'}^f$. Therefore, X_k^f satisfies Condition (3.1) and Algorithm 1 is correct.

Theorem 4 *Algorithm 1 has a time complexity of* $O(n^2)$.

Lines 1-2 of the algorithm construct the initial partition. In line 1, calculating the Γ-parameter sets has a time complexity of $O(n^2)$, and calculating the $\overline{\Gamma}$-parameter sets and γ-values has a time complexity of $O(n)$.

Lines 5-12 perform validation & merging. In this process, the number of inner and outer while loops iterations are dependent on each other. The extreme cases are:

Case 1. the initial partition has n subsets and the output partition has n subsets. In this case, the outer while loop is performed n times but the inner while loop is never performed. In this case, lines 6-8 have a constant time, therefore, validation & merging has a time complexity of $O(n)$.

Case 2. the initial partition has n subsets and the output partition has 1 subset and each time, two adjacent subsets are merged. In this case, the inner while loop is performed $n - 1$ times but the outer while loop is performed only once. Lines 10-11 have a constant time complexity, and line 12 has a time complexity of $O(n)$. Therefore, validation & merging has a time complexity of $O(n^2)$.

Finally, the finest-Condorcet partition algorithm has a time complexity of $O(n^2)$.

Example 2 *Consider the instance given in Example 1. The γ-values for this instance are*

$$\gamma_1 = 5, \gamma_2 = 3, \gamma_3 = 3, \gamma_4 = 2, \gamma_5 = 1, \gamma_6 = 0.$$

The initial partition is $\boldsymbol{X}^0 = \{\{1\}, \{2,3\}, \{4\}, \{5\}, \{6\}\}$. Here, item 1 has the highest γ-value, items 2 and 3 have the second highest value, etc. Next, set the working partition to the initial partition (i.e., $\widetilde{\boldsymbol{X}} \leftarrow \boldsymbol{X}^0$). Afterward, start validation & merging.
Iteration 1: Item 1 is pairwise preferred over all items in the higher-indexed subsets; hence, the working partition remains unchanged.
Iteration 2: Item 2 is not pairwise preferred over item 4. Hence, subsets $\{2,3\}$ and $\{4\}$ are merged to satisfy NXCC. This gives that $\widetilde{\boldsymbol{X}} = \{\{1\}, \{2,3,4\}, \{5\}, \{6\}\}$.
Iteration 3: Items 2, 3, and 4 are pairwise preferred over items 5 and 6; hence, the working partition remains unchanged.
Iteration 4: Item 5 is pairwise preferred over item 6. Therefore, we have $\boldsymbol{X}^f = \{\{1\}, \{2,3,4\}, \{5\}, \{6\}\}$.

3.2 Upper Bounding via Approximate Condorcet Partitioning

Condorcet partitioning can be very useful for expediting KEMENY-AGG, particularly when the resulting partition has many small subsets. However, some instances do not have a proper partition and, in various other cases when they do, the partition may yield relatively few subsets and/or very large subsets. Yoo and Escobedo (2021) reported that a sizeable fraction of the real-world instances with ties tested therein yielded Condorcet partitions with these disadvantageous characteristics. For these reasons, exact decomposition is useful only for a limited number of problems.

This section introduces Approximate Condorcet Partitioning (ACP), which is useful for any strict or non-strict instance of KEMENY-AGG. ACP is designed to return a partition with relatively more and smaller subsets, which is not strictly a Condorcet partition (i.e., the subset orderings may conflict with the Kemeny optimal solution(s)) but retains some of the computationally beneficial structure of this social choice-inspired concept. Formal guarantees of the resulting solutions are also derived.

Whenever $\mathbf{X}^f$ contains one or more large subsets, ACP constructs a partition of $\boldsymbol{\mathcal{X}}$ that leverages the finest-Condorcet partition, $\mathbf{X}^f$, and the initial partition, $\mathbf{X}^0$, both obtained from Algorithm 1. Recall that $\mathbf{X}^0$ is easily constructed based on the calculated parameter γ_i, defined as the number of items over which item i is pairwise preferred. Typically, $\mathbf{X}^0$ consists of many subsets, a large fraction of which are subsequently merged in Algorithm 1 to satisfy NXCC. Whenever the validation & merging step creates large subsets in $\mathbf{X}^f$, ACP builds a different item partition from $\mathbf{X}^0$, which may violate NXCC but retains some of the convenient structure of $\mathbf{X}^f$. That is, only those subsets of $\mathbf{X}^f$ which are difficult to solve by exact methods are broken down by ACP; all other subsets are left unchanged, and this preserves some of the ordered subsets. Thus, this new partition is designed to yield a higher number of computationally manageable subsets, i.e., whose KEMENY-AGG subproblems are solvable with exact methods. A key insight behind ACP is that items that have close γ-values are more likely to be close to each other in the consensus ranking; hence, smaller subsets are formed based on these calculated parameters, keeping those with similar values near one another. The pseudocode of the proposed algorithm for ACP is presented in Algorithm 2. To summarize its steps, let h be a user-specified threshold, which can be set to the maximum KEMENY-AGG instance size that is solvable to optimality within a reasonable time (i.e., based on prior findings and available computational resources). That is, for some subset $X_k^f \in \mathbf{X}^f$ with $|X_k^f| \leq h$,

Algorithm 2: Approximate Condorcet Partitioning

Input : $[c_{ij}] \in \mathbb{Z}^{n \times n}$ (CR matrix), h
Output: Approximate Condorcet Partition ($\mathbf{X}_h^{ACP}$)

$\mathbf{X}^f$, $\mathbf{X}^0 \leftarrow$ Finest-Condorcet Partition($[c_{ij}]$);
$\mathbf{X}_h^{ACP} \leftarrow \emptyset$;
for $k = 1$ **to** $|\boldsymbol{X}^f|$ **do**
    if $|\boldsymbol{X}_k^f| \leq h$ **then**
        Append $\mathbf{X}_k^f$ to $\mathbf{X}_h^{ACP}$;
    else
        Let $\{X_u^0, \ldots, X_v^0\} = X_{uv}^0$ be the set of consecutive subsets of $\mathbf{X}^0$ that have been merged together to form $\mathbf{X}_k^f$;
        $q \leftarrow u$;
        while $q \leq v - 1$ **do**
            if $|X_q^0| \geq h$ ***or*** $q = v - 1$ **then**
                Append X_q^0 to $\mathbf{X}_h^{ACP}$;
                $q \leftarrow q + 1$;
            else
                Let $l \leq v$ be the highest index such that $|X_q^0 \cup \cdots \cup X_l^0| \leq h$;
                Merge subsets $X_q^0, \ldots, X_l^0$ and append it to $\mathbf{X}_h^{ACP}$;
                $q \leftarrow l + 1$;
return $\boldsymbol{X}_h^{ACP}$

ACP skips this subset. However, if $|X_k^f| > h$, then the algorithm evaluates the corresponding adjacent subsets in $\mathbf{X}^0$, say $\{X_u^0, \ldots, X_v^0\} = X_{uv}^0$, which were merged together during the validation & merging step of Algorithm 1 to form X_k^f. In the next step, the algorithm tries to merge adjacent subsets of X_{uv}^0 as long as their combined size does not exceed h. During this process, if the size of a subset of X_{uv}^0 is already greater than h, the subset is not merged with any other subsets and is left unchanged for the remainder of the algorithm.

Let $\mathbf{X}_h^{ACP}$ denote the partition obtained using threshold h. After obtaining ACP, the ensuing steps aim to obtain a high-quality solution via ACP: 1) solve those subsets of $\mathbf{X}_h^{ACP}$ whose size is at most h to optimality, 2) for each subset whose size exceeds

h, tie all its items in the case of non-strict rankings and permute its items randomly in the case of strict rankings. Step 2 aims to find a quick solution for those subsets that are deemed difficult to solve to optimality. Similar to Condorcet Partitioning, to obtain a complete ordering of $\boldsymbol{\mathcal{X}}$, items in lower-indexed subsets of $\mathbf{X}_h^{ACP}$ are strictly ranked ahead of items in higher-indexed subsets

Remark 1 *The proposed solution method via ACP becomes an exact method if* $|X_k^f| \leq h \ \forall X_k \in \boldsymbol{X}^f$.

The ensuing small example helps illustrate ACP.

Example 3 *Consider an instance with 5 rankings of 10 items and set the threshold as* $h = 3$. *The input rankings* ($\boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^5$), *consensus ranking* ($\boldsymbol{\sigma}^*$), *and* γ*-value of each item are given by*

Item	Input Rankings					$\boldsymbol{\sigma}^*$
	$\boldsymbol{\sigma}^1$	$\boldsymbol{\sigma}^2$	$\boldsymbol{\sigma}^3$	$\boldsymbol{\sigma}^4$	$\boldsymbol{\sigma}^5$	
1	8	1	9	10	1	7
2	2	3	7	6	8	4
3	5	10	2	4	10	5
4	6	5	5	2	2	2
5	3	4	10	9	9	10
6	1	9	1	1	4	1
7	10	7	6	5	5	9
8	9	6	4	8	7	8
9	7	2	8	7	3	6
10	4	8	3	3	6	3

Item	$\boldsymbol{\gamma}$
1	3
2	5
3	4
4	8
5	1
6	9
7	2
8	2
9	4
10	7

The initial partition is $\boldsymbol{X}^0 = \{\{6\}, \{4\}, \{10\}, \{2\}, \{3, 9\}, \{1\}, \{7, 8\}, \{5\}\}$ *and the finest-Condorcet partition is* $\boldsymbol{X}^f = \{\{6\}, \{4\}, \{10\}, \{1, 2, 3, 5, 7, 8, 9\}\}$. *The ACP algorithm leaves subsets* $\{6\}, \{4\}$ *and* $\{10\}$ *unchanged, as their size is less than* h, *but it seeks to further decompose the fourth subset of* $\boldsymbol{X}^f$ *whose size exceeds* h. *Note that, subsets* $\{2\}, \{3, 9\}, \{1\}, \{7, 8\}, \{5\} \in \boldsymbol{X}^0$ *were merged in the validation & merging to*

form subset $\{1,2,3,5,7,8,9\}$. *ACP proceeds by merging subsets* $\{2\},\{3,9\}$ *to form subset* $\{2,3,9\}$ *whose size reaches* h*; subsets* $\{1\},\{7,8\}$ *are merged to form subset* $\{1,7,8\}$ *whose size also reaches the threshold; and subset* $\{5\}$ *is left unchanged. Therefore, the output ACP is given by* $\boldsymbol{X}_{h=3}^{ACP} = \{\{6\},\{4\},\{10\},\{2,3,9\},\{1,7,8\},\{5\}\}$. *Afterward, one* KEMENY-AGG *subproblem is solved for each subset and their respective solutions are concatenated (for completeness, the concatenated subproblem solutions matches the optimal solution of the full problem).*

3.2.1 Provable Guarantees from Partitioning

This subsection derives three different approximation factors, all of which are easy to calculate and specific to the characteristics of an instance. The first of these is applicable to any item-partitioning scheme, including those that may not be consistent with Condorcet properties—e.g., see Aledo *et al.* (2021) for a decomposition based on Borda scores and Liu *et al.* (2021) for a hierarchical clustering method. The second and third derived approximation factors provide improved guarantees for the ACP solution, for non-strict and strict ranking instances, respectively. For the remainder of this section, let LB be a lower bound on d_{KS}^*.

Lemma 1 *Let* $\boldsymbol{X} = \{X_1, X_2, \ldots, X_w\}$ *be any given partition of* X *and* $\hat{\boldsymbol{\sigma}}$ *be a complete ranking obtained by independently solving the subsets of* $\boldsymbol{X}$ *and concatenating the solutions of these subproblems. If* $d_{KS}(\hat{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*)$ *is bounded by a constant* β*, the complete ranking* $\hat{\boldsymbol{\sigma}}$ *is an* $(1+\alpha)$*-approximate solution, where* $\alpha = \beta/LB$.

$$d_{KS}(\hat{\boldsymbol{\sigma}}) \leq\ d_{KS}(\boldsymbol{\sigma}^*) + \beta = d_{KS}(\boldsymbol{\sigma}^*) + \alpha LB \leq d_{KS}(\boldsymbol{\sigma}^*) + \alpha d_{KS}(\boldsymbol{\sigma}^*) = (1+\alpha) d_{KS}(\boldsymbol{\sigma}^*).$$

Lemma 2 *Let* $\boldsymbol{X} = \{X_1, X_2, \ldots, X_w\}$ *be any given partition of* $\boldsymbol{\mathcal{X}}$ *and* $\hat{\boldsymbol{\sigma}}$ *be a complete ranking obtained by independently solving the subsets of* $\boldsymbol{X}$ *(using any method*

of choice) and concatenating the solutions of these subproblems. The term $d_{KS}(\hat{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*)$ *is bounded by*

$$2\sum_{k=1}^{w-1}\sum_{k=k+1}^{w}\sum_{i\in X_k}\sum_{j\in X_{k'}}\max\left(0,\,(2s_{ji}+t_{ij})-(2s_{ij}+t_{ij}),\,(2s_{ji}+t_{ij})-(s_{ij}+s_{ji})\right)+$$
$$\sum_{k=1}^{w}\sum_{(i,j)\in X_k}\max\left(2s_{ij}+t_{ij},\,2s_{ji}+t_{ij},\,s_{ij}+s_{ji}\right)-\min\left(2s_{ij}+t_{ij},\,2s_{ji}+t_{ij},\,s_{ij}+s_{ji}\right). \tag{3.2}$$

Consider an item pair (i, j), where $i \in X_k$, $j \in X_{k'}$, $k < k'$ (items from different subsets); since $\hat{\sigma}_i < \hat{\sigma}_j$, the contribution of this pair in $d_{KS}(\hat{\boldsymbol{\sigma}})$ is $2s_{ji} + t_{ij}$, while the contribution of this pair in $d_{KS}(\boldsymbol{\sigma}^*)$ must be exactly one of $2s_{ji} + t_{ij}$, $2s_{ij} + t_{ij}$, or $s_{ij} + s_{ji}$. Therefore, the additional distance accrued by (i, j) in $d_{KS}(\hat{\boldsymbol{\sigma}})$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$\max\Big(0,\,(2s_{ji}+t_{ij})-(2s_{ij}+t_{ij}),\,(2s_{ji}+t_{ij})-(s_{ij}+s_{ji})\Big).$$

Consider a pair of distinct items (i, j), where $i, j \in X_k$ (items within the same subset); the additional distance accrued by (i, j) in $d_{KS}(\hat{\boldsymbol{\sigma}})$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$\max\Big(2s_{ij}+t_{ij},\,2s_{ji}+t_{ij},\,s_{ij}+s_{ji}\Big)-\min\Big(2s_{ij}+t_{ij},\,2s_{ji}+t_{ij},\,s_{ij}+s_{ji}\Big). \tag{3.3}$$

Since the exact orderings of i and j in $\boldsymbol{\sigma}^*$ and $\hat{\boldsymbol{\sigma}}$ are not yet known, Eq. (3.3) considers the worst case. Expressly, the contribution of (i, j) in $d_{KS}(\boldsymbol{\sigma}^*)$ is taken as the smallest of the three possible values of $d_{KS}(\sigma_{ij})$, whereas the contribution of this pair in $d_{KS}(\hat{\boldsymbol{\sigma}})$ is taken as the largest of the three possible values of $d_{KS}(\sigma_{ij})$. Finally, the right-hand side of Eq. (3.2) has been multiplied by 2 since d_{KS} counts each item-pair twice.

Combining Lemmas 1 and 2 provides a formal guarantee of the solution quality of an arbitrary partition of X. The approximation factor holds regardless of how the items within each subset in the partition are ordered (i.e., it is a worst-case bound), and thus any method of choice can be used. As such, the quality of the solution is improved by determining orderings that more closely align with the optimal solution. The next two theorems derive a tighter guarantee by leveraging the specific solution methods for solving the subsets of $\mathbf{X}_h^{ACP}$.

Theorem 5 *Assume that the input rankings are non-strict and let* $\boldsymbol{X}_h^{ACP} = \{X_1, \ldots, X_w\}$ *be the ACP partition obtained using threshold* h*. Let* $\hat{\boldsymbol{\sigma}}$ *be the complete ranking obtained via ACP from the following two steps: 1) solve subsets of at most size* h *to optimality, 2) tie all items in subsets of size greater than* h*. The term* $d_{KS}(\hat{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*)$ *is bounded by*

$$2\sum_{k=1}^{w-1}\sum_{k=k+1}^{w}\sum_{i\in X_k}\sum_{j\in X_{k'}} \max\left(0, (2s_{ji}+t_{ij}) - (2s_{ij}+t_{ij}), (2s_{ji}+t_{ij}) - (s_{ij}+s_{ji})\right) + \sum_{k=1}^{w}\sum_{(i,j)\in X_k} \big((s_{ij}+s_{ji}) - \min\left(2s_{ij}+t_{ij}, 2s_{ji}+t_{ij}, s_{ij}+s_{ji}\right) \big). \tag{3.4}$$

Let $\overline{\boldsymbol{\sigma}}$ be an auxiliary ranking obtained from $\mathbf{X}_h^{ACP}$, whereby all items in each subset $X_k \in \mathbf{X}_h^{ACP}$ are tied $\forall k \in \{1, \ldots, w\}$ (items in lower-indexed subsets remain ranked ahead of items in higher-indexed subsets). Since $\hat{\boldsymbol{\sigma}}$ solves all the subsets whose size is at most h to optimality and ties all the items in subsets whose size is greater than h, we have that $d_{KS}(\hat{\boldsymbol{\sigma}}) \leq d_{KS}(\overline{\boldsymbol{\sigma}})$. Furthermore, we show that $d_{KS}(\overline{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*) \leq \beta$.

Consider an item pair (i, j), where $i \in X_k$, $j \in X_{k'}$, $k < k'$ (items from different subsets); since $\overline{\sigma}_i < \overline{\sigma}_j$, the contribution of this pair in $d_{KS}(\overline{\boldsymbol{\sigma}})$ is $2s_{ji} + t_{ij}$, while the contribution of this pair in $d_{KS}(\boldsymbol{\sigma}^*)$ must be exactly one of $2s_{ji} + t_{ij}$, $2s_{ij} + t_{ij}$, or

$s_{ij} + s_{ji}$. Therefore, the additional distance accrued by (i, j) in $d_{KS}(\overline{\boldsymbol{\sigma}})$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$\max\left(0,\, (2s_{ji} + t_{ij}) - (2s_{ij} + t_{ij}),\, (2s_{ji} + t_{ij}) - (s_{ij} + s_{ji})\right).$$

Consider a pair of distinct items (i, j), where $i, j \in X_k$ (items within the same subset); the additional distance accrued by (i, j) in $\overline{\boldsymbol{\sigma}}$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$(s_{ij} + s_{ji}) - \min\left(2s_{ij} + t_{ij},\, 2s_{ji} + t_{ij},\, s_{ij} + s_{ji}\right). \tag{3.5}$$

Since the exact orderings of i and j in $\boldsymbol{\sigma}^*$ are not yet known, Eq. (3.5) considers the worst case. Expressly, the contribution of (i, j) in $d_{KS}(\boldsymbol{\sigma}^*)$ is taken as the smallest of the three possible values of $d_{KS}(\sigma_{ij})$, whereas the contribution of this pair in $d_{KS}(\hat{\boldsymbol{\sigma}})$ is $(s_{ij} + s_{ji})$ as $\overline{\boldsymbol{\sigma}}$ ties all items within each subset. Finally, the first term of Eq. (3.4) has been multiplied by 2 since d_{KS} counts each item-pair twice.

Theorem 6 *Assume that the input rankings are strict and let* $\boldsymbol{X}_h^{ACP} = \{X_1, \ldots, X_w\}$ *be the ACP partition obtained using threshold* h*. Let* $\hat{\boldsymbol{\sigma}}$ *be the complete ranking obtained via ACP from the following two steps: 1) solve subsets of at most size* h *to optimality, 2) randomly permute items within subsets of size greater than* h*. The term* $d_{KS}(\hat{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*)$ *is bounded by*

$$4\sum_{k=1}^{w-1}\sum_{k=k+1}^{w}\sum_{i\in X_k}\sum_{j\in X_{k'}}\max\left(0,\, s_{ji} - s_{ij}\right) + 2\sum_{k=1}^{w}\sum_{(i,j)\in X_k}\left(s_{ji}\mathbb{1}_{\overline{\sigma}_i<\overline{\sigma}_j} + s_{ij}\mathbb{1}_{\overline{\sigma}_j<\overline{\sigma}_i} - \min\left(s_{ij},\, s_{ji}\right)\right); \tag{3.6}$$

where $\overline{\boldsymbol{\sigma}}$ *is an auxiliary ranking obtained by randomly permuting all items in subset* X_k*,* $\forall k \in \{1, \ldots, w\}$ *(while ranking items in the lower-indexed subsets strictly ahead of items in the higher-indexed subsets), and* $\mathbb{1}$ *is an indicator function.*

Order the items within subset $X_k \in \mathbf{X}_h^{ACP}$ whose size is greater than h in $\hat{\boldsymbol{\sigma}}$ the same as in $\overline{\boldsymbol{\sigma}}$. Since $\hat{\boldsymbol{\sigma}}$ solves all subsets whose size is at most h to optimality whereas $\hat{\boldsymbol{\sigma}}$ randomly permutes these items, we have that $d_{KS}(\hat{\boldsymbol{\sigma}}) \leq d_{KS}(\overline{\boldsymbol{\sigma}})$. Next, we show that $d_{KS}(\overline{\boldsymbol{\sigma}}) - d_{KS}(\boldsymbol{\sigma}^*) \leq \beta$.

Consider an item pair (i, j), where $i \in X_k$, $j \in X_{k'}$, $k < k'$ (items from different subsets); since $\overline{\sigma}_i < \overline{\sigma}_j$, the contribution of this pair in $d_{KS}(\overline{\boldsymbol{\sigma}})$ is $2s_{ji}$, while the contribution of this pair in $d_{KS}(\boldsymbol{\sigma}^*)$ must be exactly one of $2s_{ji}$ or $2s_{ij}$ (when the input rankings are strict, $d_{KS}(\sigma_{ij})$ has only two possible values since the aggregate ranking is assumed to be strict as well). Therefore, the additional distance accrued by (i, j) in $d_{KS}(\overline{\boldsymbol{\sigma}})$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$\max\left(0,\, (2s_{ji} - 2s_{ij})\right).$$

Consider a pair of distinct items (i, j), where $i, j \in X_k$ (items within the same subset); the additional distance accrued by (i, j) in $\overline{\boldsymbol{\sigma}}$ relative to $d_{KS}(\boldsymbol{\sigma}^*)$ is at most

$$2s_{ji}\mathbb{1}_{\overline{\sigma}_i < \overline{\sigma}_j} + 2s_{ij}\mathbb{1}_{\overline{\sigma}_j < \overline{\sigma}_i} - \min\left(2s_{ij},\, 2s_{ji}\right). \tag{3.7}$$

Since the exact orderings of i and j in $\boldsymbol{\sigma}^*$ is not yet known, Eq. (3.7) considers the worst case. Expressly, the contribution of (i, j) in $d_{KS}(\boldsymbol{\sigma}^*)$ is taken as the smallest of the three possible values of $d_{KS}(\sigma_{ij})$, whereas the contribution of this pair in $d_{KS}(\hat{\boldsymbol{\sigma}})$ is is $2s_{ji}$ if $\overline{\sigma}_i < \overline{\sigma}_j$, and $2s_{ij}$ otherwise. Finally, the first term of Eq. (3.6) has been multiplied by 2 since d_{KS} counts each item-pair twice. The approximation factors are computed after a partition is obtained, meaning they are instance-specific and not constant; their value becomes relatively small when the given partition aligns well with the structure of the consensus ranking. ACP offers significant advantages over

various other partitioning methods in this regard. Furthermore, it uses the calculated γ-parameters used to obtain $\mathbf{X}^0$ to reduce the number of rank reversals between items across many more subsets than are contained in $\mathbf{X}^f$. It leverages structural information from $\mathbf{X}^f$ by retaining item-ordering of subsets that are relatively easy to solve and their relative ordering to other subsets. It is important to emphasize that, while the derived approximation factors provide a guarantee of the solution quality, we are interested in partitions that tend to produce high-quality solutions rather than those that minimize the approximation factor. Indeed, by increasing the number of subsets (i.e., reducing the value of h), one may decrease their values; however, doing so can also negatively impact the resulting solution, as increasing the number of subsets can be viewed as placing more constraints on the ordering of certain items.

Lemma 1, Theorem 5, and Theorem 6 require a lower bound on KEMENY-AGG. The lower bound using pairwise information (presented later in the chapter) is used in the computational results. This lower bound equals zero if and only if all input rankings are identical, which would render the approximation factors incomputable; since this special case does not require analysis of any kind (i.e., the consensus ranking equals the unanimous ranking), this does not pose a serious issue for the proposed approximation algorithm.

3.2.2 Computational Comparisons

This section compares ACP with some of the prominent approximation algorithms mentioned in Chapter 1. The selected methods for instances with strict rankings are the proposed solution method via ACP, BestInput, DeterministicKwikSort, KwikSort, LPKwikSort, and Spearman's footrule; and for instances with non-strict rankings, they are the proposed solution method via ACP, BestInput, and BucketPivot (using the algorithm's default parameter). Note that not all algorithms mentioned in

Chapter 1 are tested on all instances, as some are not designed to handle non-strict rankings. Specifically, RepeatChoice and LPKwikSort$_h$ are not applied to such instances, as they restrict the consensus ranking to be strict, which does not align with the more general assumption that the output rankings may also be non-strict. Note that when the input rankings are strict, RepeatChoice becomes KwikSort.

We use two real-world data sets. The first data set is from Cohen-Boulakia *et al.* (2011) and is henceforth denoted as the Biomedical data set. Each instance contains four non-strict rankings of genes possibly associated with Breast Cancer, Prostate Cancer, Bladder Cancer, Neuroblastoma, Retinoblastoma, ADHD (Attention Deficit Hyperactivity Disorder), and LQTS (Long QT Syndrome). Each set of input rankings is the result of querying for the respective diseases in biological databases using four different methods. The objective of the referenced study is to reduce the variability of information retrieval techniques by consolidating their outputs. The second data set consists of instances with and without ties from Preflib (Mattei and Walsh, 2013), a library of preference data; namely instances from "TOC - Orders with Ties - Complete List" and "SOC - Strict Orders - Complete List" with over 100 items [1] .

All experiments were carried out on a PC with an Intel(R) Xeon(R) CPU E5-2680 @ 2.40 GHz with 64 GB RAM. All KEMENY-AGG subproblems were solved using the exact binary programming formulation of Yoo and Escobedo (2021) via CPLEX solver version 12.10.0. The Spearman's footrule rank aggregation problem was solved via minimum cost perfect matching in bipartite graphs (Dwork *et al.*, 2001). For ACP, we tested thresholds $h = 30, 40, 50$. The experimental results report, for each instance, number of items (n), number of input rankings (m), size of the largest subset of $\mathbf{X}^f$ ($(X_l^f)^{max}$), run-time (Time) and relative optimality gap (Gap %) attained by

[1] The Preflib data set presents the input rankings in the form of sorted lists, but a few items repeat in certain instances, presumably due to error. To overcome this issue, we adjusted these instances by keeping the first appearance of each item in each list and deleting any extra occurrences

each tested method. Run-times include pre-processing time required by each specific method. The relative optimality gap for each method is calculated as the difference between its objective value and the lower bound (Eq. (3.9)), divided by its objective value. It is displayed as a percentage, for convenience (relative optimality gap, multiplied by 100); the best relative optimality gap % attained for each instance is shown in bold. For ACP, the tested threshold value (h) and calculated approximation factor (AF) are reported. Lastly, for each data set, the average relative optimality gap and the geometric mean of run-times achieved by each of the selected algorithms are displayed. All statistics are rounded to two decimal points.

Table 3.1 reports the computational results for the Biomedical data set. These results exclude the ADHD and LSQT instances since they were both relatively small and could be easily solved to optimality without partitioning. By comparing n and $(X_l^f)^{max}$ columns (the number of items and the size of the largest subset of $\mathbf{X}^f$) , we can see that the finest-Condorcet partition over all instances is either null or it contains a rather large subset that is difficult to solve to optimality. Overall, none of the selected methods had a dominant performance on all of these instances. On average, BestInput had the lowest relative optimality gap, ACP the second lowest, and BucketPivot the third lowest. As Table 3.1 shows, BestInput had a top-2 performance in terms of solution quality for all 5 instances, ACP for 3 out of 5 instances, and BucketPivot for 2 out of 5 instances. Among the three tested threshold values for ACP, $h = 50$ achieved the best solution quality, but it also had higher run-times and approximation factor. The average approximation factor achieved over these instances was 1.14. Finally, ACP yielded near-optimal solutions (with gaps of up to 0.11%) on Bladder Cancer and Retinoblastoma for which each $\mathbf{X}^f$ contained multiple subsets of small-to-medium sizes.

Table 3.2 reports the results of the Preflib TOC data set. The general character-

istics of $\mathbf{X}^f$ were the same as for the Biomedical data set. Overall, ACP exhibited a dominant performance, achieving the lowest relative optimality gap for all 85 instances. In fact, its worst optimality gap over all instances and three tested thresholds of 3.37% was lower than the best optimality gap achieved by either BucketPivot or BestInput. Most impressively, the average approximation factor was 1.02, further highlighting the comparative robustness of ACP. It is important to remark that the best approximation factor for the methods that allow ties in the output ranking is 9. To round out the results, BucketPivot was faster but yielded a lower solution quality than BestInput. BucketPivot had an optimality gap of up to 97.92% and BestInput of up to 41.08%. BestInput did not terminate after 1 hour of run-time for instances #77-#80, likely owing to its quadratic complexity with respect to both the number of items and the number of input rankings. These four instances are much larger than the rest: #77-#78 have 5,000 input rankings and #79-#80 have at least 379 items and at least 556 input rankings.

Table 3.3 reports the results of the Preflib SOC data set. The general characteristics of $\mathbf{X}^f$ were the same as for the TOC and the Biomedical data sets. Since all of the item methods tested for this data set output a strict ranking, the output ranking of ACP was forced to be strict as well. While this restriction does not take full advantage of its intended purpose, ACP still exhibited a very good performance, headlined by its average approximation factor of 1.06 over this data set. While LP-KwikSort dominated in solution quality, achieving the lowest optimality gap in all but 2 of the 22 instances, it also had relatively high run-times of over 10 minutes—in fact, its lowest run-time was greater than the highest run-time attained by all other methods. This is due to fact that LPKwikSort requires solving an LP with $O(n^3)$ constraints, which causes memory issues for large instances. Conversely, ACP produced competitive solutions in far less time (it solved each SOC instance in under 30

seconds). In the 20 instances where LPKwikSort had the lowest optimality gap, ACP had the second lowest in 13 instances and BestInput in 7 instances; however, the worst relative optimality gap of BestInput (13%) was much higher than ACP's (not more than 3.91%). Furthermore, apart from instances #1 and #5 (for which ACP achieved the lowest optimality gap), the relative optimality gaps attained by LPKwikSort and ACP were very close, differing by no more than 2.8 percentage points. To round out the results, DeterministicKwikSort produced neither high-quality solutions nor low run-times. While KwikSort had quick run-times, they were similar to those of BestInput and Spearman's footrule, which yielded better solutions.

All things considered, BestInput had a good performance on the Biomedical and SOC data sets, but performed poorly on the TOC data set, especially when the number of input rankings was high. LPKwikSort had an excellent performance on strict rankings, but its run-time increases very fast with n, which makes it unattractive for large-scale problems. Additionally, this method is only able to handle strict rankings, and its non-strict variant, LPKwikSort$_h$, does not allow the consensus ranking to include ties, thereby limiting its general application. ACP has a very good performance on the Biomedical and SOC data sets and a dominant performance on the TOC data set. Overall, it had a very robust performance in terms of solution quality and run-times on both strict and non-strict rankings instances. Quite remarkably, none of the tested instances of up to 2,820 items exceeded 90 seconds in run-time, which includes the time to construct $\mathbf{X}^f$ and $\mathbf{X}^{ACP}$ and to solve the corresponding KEMENY-AGG subproblems for all ACP subsets whose size is under threshold h. In fact, the time to calculate the CR matrix, to construct $\mathbf{X}^f$ and $\mathbf{X}^{ACP}$, and to calculate the respective approximation factor took less than 1 second for each instance of the Biomedical and SOC data sets and less than 12 seconds for each instance of the TOC data set. As a final note, it is important to highlight that although the approximation factors

achieved by ACP are instance-specific, they are considerably lower for all 112 tested instances than the guarantees offered by any existing constant-factor approximation algorithm for KEMENY-AGG. Indeed, the worst ACP approximation factor obtained was 1.3.

Table 3.1: Performance Metrics of the Selected Approximation Algorithms for Solving Biomedical Instances

Instance	n	m	$(X_l^f)^{max}$	BucketPivot		BestInput		Approximate Condorcet Partitioning								
								h=30			h=40			h=50		
				Gap %	Time	Gap %	Time	Gap %	Time	AF	Gap %	Time	AF	Gap %	Time	AF
Bladder Cancer	308	4	266	14.19	0.11	**0.00**	0.45	0.01	0.95	1.01	0.01	0.85	1.01	0.01	2.19	1.01
Breast Cancer	386	4	386	25.91	0.18	**5.05**	0.67	22.55	2.72	1.29	22.67	4.78	1.29	22.79	6.39	1.3
Neuroblastoma	431	4	431	13.63	0.18	5.15	0.82	4.06	1.4	1.05	3.76	4.55	1.04	**3.62**	7.74	1.04
Prostate Cancer	218	4	216	29.13	0.05	11.35	0.21	18.69	1.07	1.23	18.30	1.92	1.22	18.26	2.28	1.22
Retinoblastoma	402	4	358	0.87	0.18	0.19	0.74	**0.11**	0.51	1.01	**0.11**	1.6	1.01	**0.11**	2.55	1.01
Average				16.75	0.13	4.35	0.52	9.08	1.15	1.12	8.97	2.24	1.12	8.96	3.63	1.14

Table 3.2: Performance Metrics of the Selected Approximation Algorithms for Solving Toc Instances with More than 100 Items

Instance	n	m	$X^{If})^{max}$	BucketPivot		BestInput		Approximate Condorcet Partitioning								
								$h = 30$			$h = 40$			$h = 50$		
				(%) Gap	Time	(%) Gap	Time	(%) Gap	Time	AF	(%) Gap	Time	AF	(%) Gap	Time	AF
ED-14-02	100	5,000	100	0	2.48	–	$\geq 3,600$	**0.00**	2.5	1.00	**0.00**	2.5	1.00	**0.00**	2.5	1.00
ED14-03	100	5,000	100	0	2.17	–	$\geq 3,600$	**0.00**	2.17	1.00	**0.00**	2.17	1.00	**0.00**	2.19	1.00
MD-03-03	102	32	102	0	0.01	34.24	4.07	**0.00**	0.02	1.00	**0.00**	0.02	1.00	**0.00**	0.02	1.00
MD-03-05	103	31	103	0	0.02	40.94	4.85	**0.00**	0.02	1.00	**0.00**	0.02	1.00	**0.00**	0.02	1.00
MD-03-06	133	38	133	0	0.02	41.08	11.56	**0.00**	0.02	1.00	**0.00**	0.02	1.00	**0.00**	0.02	1.00
MD-03-08	147	51	147	0	0.02	36.54	25.17	**0.00**	0.03	1.00	**0.00**	0.03	1.00	**0.00**	0.05	1.00
MD-03-07	155	51	155	0	0.03	41.08	20.96	**0.00**	0.06	1.00	**0.00**	0.06	1.00	**0.00**	0.06	1.00
ED-10-50	170	4	170	26.81	0.02	12.41	0.14	**1.42**	1.48	1.07	1.55	1.86	1.10	1.45	3.2	1.13
ED-10-49	351	4	351	13.34	0.30	14.22	0.77	0.74	2.85	1.03	0.74	4.19	1.04	**0.65**	7.32	1.04
ED-18-01	379	723	379	98.84	0.41	–	$\geq 3,600$	**0.00**	0.44	1.00	**0.00**	0.45	1.00	**0.00**	0.42	1.00
ED-18-03	477	556	476	99.36	0.45	–	$\geq 3,600$	**0.00**	0.48	1.00	**0.00**	0.48	1.00	**0.00**	0.47	1.00
ED-11-12	1,210	4	1,207	10.05	1.27	4.70	6.36	2.73	6.13	1.03	2.71	9.24	1.03	**2.70**	13.44	1.03
ED-11-31	1,223	4	1,223	14.30	1.16	8.07	6.58	1.70	5.77	1.02	1.69	8.63	1.02	**1.66**	12.24	1.02
ED-11-09	1,272	4	1,272	18.40	1.52	7.79	6.85	3.37	8.59	1.04	3.34	13.83	1.04	**3.28**	20.90	1.04
ED-11-23	1,342	4	1,341	9.88	1.39	8.58	7.98	1.65	6.77	1.02	1.62	10.72	1.02	**1.59**	15.36	1.02
ED-11-21	1,347	4	1,347	9.32	1.56	4.98	7.69	2.64	8.16	1.03	2.61	12.47	1.03	**2.59**	19.13	1.03
ED-11-37	1,351	4	1,351	14.09	1.66	4.33	8.11	3.05	9.86	1.04	2.99	15.83	1.04	**2.95**	24.15	1.04
ED-11-25	1,356	4	1,353	15.42	1.55	8.07	7.89	2.33	8.56	1.03	2.26	14.05	1.03	**2.22**	19.42	1.03
ED-11-13	1,363	4	1,363	4.79	1.47	6.18	8.2	1.60	6.44	1.02	1.58	9.38	1.02	**1.57**	13.38	1.02
ED-11-29	1,368	4	1,368	20.58	1.75	3.92	8.55	3.29	10.20	1.04	3.26	16.67	1.04	**3.20**	25.94	1.04
ED-11-14	1,375	4	1,372	21.44	1.77	3.73	8.36	3.21	9.89	1.04	3.14	16.03	1.04	**3.10**	23.74	1.04
ED-11-30	1,386	4	1,384	4.73	1.64	5.89	8.78	2.40	8.85	1.03	2.35	14.41	1.03	**2.32**	21.21	1.03
ED-11-06	1,449	4	1,449	15.03	1.77	4.96	10.02	2.72	9.69	1.03	2.68	14.52	1.03	**2.64**	22.66	1.03
ED-11-04	1,467	4	1,463	32.45	1.91	4.71	10.63	2.57	8.85	1.03	2.54	13.3	1.03	**2.52**	19.86	1.03
ED-11-07	1,474	4	1,470	3.59	1.73	7.89	10.19	1.81	7.3	1.02	1.79	10.61	1.02	**1.78**	15.7	1.02
ED-11-34	1,509	4	1,509	11.44	1.77	6.49	10.02	1.52	7.36	1.03	1.49	10.68	1.03	**1.48**	14.91	1.03
ED-11-22	1,514	4	1,513	15.78	1.86	5.59	10.09	1.86	9.52	1.02	1.82	14.7	1.02	**1.80**	20.88	1.02
ED-11-11	1,545	4	1,542	28.0	2.08	5.44	10.36	2.06	10.91	1.02	2.00	17.04	1.02	**1.97**	25.27	1.03
ED-11-15	1,563	4	1,560	27.96	1.97	6.05	10.74	2.02	9.36	1.02	1.99	14.82	1.02	**1.97**	20.75	1.02
ED-11-08	1,572	4	1,569	6.93	1.94	6.37	11.73	1.26	6.75	1.02	1.25	9.31	1.02	**1.24**	13.25	1.02
ED-11-28	1,616	4	1,611	2.69	1.89	12.57	12.05	0.78	6.72	1.01	**0.77**	9.0	1.01	**0.77**	12.38	1.01
ED-11-40	1,623	4	1,623	26.22	1.59	16.09	11.88	0.33	6.43	1.01	**0.32**	10.49	1.01	**0.32**	11.64	1.02
ED-11-36	1,634	4	1,632	22.43	2.06	7.52	11.85	1.53	9.03	1.02	1.50	13.19	1.02	**1.49**	18.36	1.02
ED-11-33	1,646	4	1,644	9.06	2.3	5.50	12.5	1.90	10.35	1.02	1.86	15.31	1.02	**1.82**	22.41	1.02
ED-11-05	1,673	4	1,672	3.91	1.98	18.91	13.52	0.66	9.91	1.01	0.64	17.89	1.01	**0.63**	88.36	1.01
ED-11-18	1,681	4	1,676	10.98	2.22	5.87	12.14	1.94	8.85	1.02	1.93	13.56	1.02	**1.90**	18.66	1.02
ED-11-16	1,708	4	1,707	6.48	2.19	7.47	13.07	1.18	8.73	1.02	**1.16**	12.64	1.02	**1.16**	17.61	1.02
ED-11-32	1,751	4	1,751	8.63	2.3	5.28	13.1	1.68	8.92	1.02	**1.65**	12.56	1.02	**1.65**	17.69	1.02
ED-11-38	1,754	4	1,752	12.13	2.36	5.69	14.16	1.98	9.06	1.02	1.95	12.83	1.02	**1.93**	17.92	1.02
ED-11-39	1,788	4	1,788	26.11	2.2	15.39	13.75	0.39	9.22	1.01	0.37	18.8	1.01	**0.35**	24.71	1.01
ED-11-68	1,826	4	1,826	31.41	2.14	9.27	14.36	**0.62**	6.3	1.01	**0.62**	7.74	1.01	**0.62**	9.11	1.01
ED-11-49	1,845	4	1,844	2.86	2.6	6.28	14.74	0.90	6.56	1.01	**0.89**	8.1	1.01	**0.89**	9.17	1.01
ED-11-20	1,870	4	1,866	2.91	2.77	6.99	15.13	1.53	11.7	1.02	1.49	17.33	1.02	**1.45**	24.35	1.02
ED-11-26	1,931	4	1,930	6.9	2.80	5.98	16.63	1.37	11.03	1.02	1.34	16.85	1.02	**1.31**	23.24	1.02
ED-11-35	1,936	4	1,935	14.67	2.80	5.88	16.69	1.56	11.02	1.02	1.54	15.58	1.02	**1.51**	21.99	1.02
ED-11-74	1,976	4	1,976	5.43	2.92	5.6	17.89	1.15	9.91	1.02	1.13	13.7	1.02	**1.12**	18.69	1.02
ED-11-60	1,977	4	1,976	3.57	2.69	9.89	16.94	**0.86**	7.94	1.01	**0.86**	10.22	1.01	**0.86**	12.66	1.01
ED-11-58	2,011	4	2,010	3.32	2.94	6.77	17.28	1.09	8.95	1.02	**1.08**	11.77	1.02	**1.08**	15.36	1.02
ED-11-62	2,014	4	2,013	3.88	2.86	11.10	17.50	0.88	9.53	1.01	0.86	12.7	1.01	**0.85**	16.69	1.01
ED-11-17	2,015	4	2,014	12.94	3.08	5.69	17.68	1.29	11.49	1.03	1.26	16.78	1.03	**1.24**	22.6	1.03
ED-11-66	2,024	4	2,024	7.13	3.22	4.86	18.31	1.97	11.64	1.02	1.95	16.32	1.03	**1.94**	22.28	1.03
ED-11-24	2,049	4	2,049	5.92	3.11	5.44	18.28	1.42	11.92	1.03	1.40	16.83	1.03	**1.39**	23.02	1.03
ED-11-67	2,066	4	2,066	7.11	3.16	7.01	18.43	1.29	10.17	1.02	**1.28**	14.09	1.03	**1.28**	19.19	1.02
ED-11-27	2,092	4	2,088	18.22	3.11	6.82	19.39	1.12	9.92	1.02	**1.1**	13.46	1.02	**1.1**	17.97	1.02
ED-11-10	2,096	4	2,095	5.82	2.95	12.00	19.08	0.65	10.53	1.01	0.63	13.88	1.01	**0.62**	19.52	1.01
ED-11-19	2,104	4	2,102	9.51	3.25	6.16	19.24	1.26	12.38	1.02	1.24	17.78	1.02	**1.22**	25.38	1.02
ED-11-50	2,111	4	2,111	1.79	3.01	7.57	20.30	0.62	7.95	1.01	**0.61**	9.12	1.01	**0.61**	10.89	1.01
ED-11-51	2,112	4	2112	6.37	3.28	4.69	19.6	1.52	10.81	1.02	1.50	14.85	1.02	**1.49**	19.92	1.02
ED-11-65	2,119	4	2,118	3.03	2.84	17.50	19.91	0.28	7.30	1.00	**0.27**	8.92	1.00	0.28	9.66	1.00
ED-11-41	2,123	4	2,123	7.78	3.28	6.98	19.6	1.04	9.03	1.01	**1.03**	11.45	1.01	**1.03**	14.25	1.02
ED-11-71	2,127	4	2,127	15.45	3.13	8.22	19.55	0.52	8.10	1.01	**0.51**	9.41	1.01	**0.51**	12.25	1.01
ED-11-46	2,133	4	2,133	4.61	3.22	9.35	20.11	1.35	9.96	1.02	**1.34**	12.92	1.02	**1.34**	16.47	1.02
ED-11-43	2,153	4	2,153	5.04	3.3	6.38	20.41	1.27	9.88	1.02	1.26	12.88	1.02	**1.25**	16.36	1.02
ED-11-48	2,194	4	2,194	8.6	3.64	10.28	20.86	1.14	12.08	1.02	1.13	16.55	1.02	**1.11**	21.72	1.02
ED-11-52	2,242	4	2,239	10.15	3.47	7.88	21.67	0.63	10.24	1.01	0.63	13.36	1.01	**0.62**	16.50	1.01
ED-11-73	2,258	4	2,257	6.07	3.47	10.51	22.06	0.30	9.34	1.00	0.30	11.36	1.00	**0.29**	13.49	1.00
ED-11-45	2,265	4	2,264	0.43	3.33	10.41	22.50	**0.07**	7.25	1.00	**0.07**	7.39	1.00	**0.07**	7.25	1.00
ED-11-70	2,276	4	2,274	3.18	3.61	7.79	22.40	0.50	10.06	1.01	0.50	12.16	1.01	**0.49**	15.48	1.01
ED-11-59	2,281	4	2,280	7.89	3.58	8.02	23.10	0.70	10.28	1.01	0.70	12.97	1.01	**0.68**	16.75	1.01
ED-11-77	2,317	4	2,317	0.10	2.88	22.13	25.64	0.09	8.13	1.00	**0.08**	8.72	1.00	**0.08**	10.22	1.00
ED-11-53	2,321	4	2,320	0.83	3.7	9.41	24.0	0.29	9.89	1.01	**0.28**	12.28	1.01	**0.28**	14.63	1.01

Instance	n	m	$Xl^f)^{max}$	BucketPivot		BestInput		Approximate Condorcet Partitioning								
								$h = 30$			$h = 40$			$h = 50$		
				(%) Gap	Time	(%) Gap	Time	(%) Gap	Time	AF	(%) Gap	Time	AF	(%) Gap	Time	AF
ED-11-69	2,338	4	2,338	4.4	3.68	6.83	25.81	0.53	11.55	1.01	0.53	10.69	1.01	**0.52**	13.77	1.01
ED-11-55	2,353	4	2,350	2.05	3.69	7.81	23.82	**0.59**	9.02	1.01	**0.59**	10.63	1.01	**0.59**	11.57	1.01
ED-11-75	2,391	4	2,390	23.09	3.86	6.90	25.1	**0.75**	9.66	1.01	**0.75**	10.80	1.01	**0.75**	12.64	1.01
ED-11-44	2,434	4	2,430	3.44	3.95	8.61	25.66	0.56	9.83	1.01	0.56	11.52	1.01	**0.55**	14.72	1.01
ED-11-64	2,446	4	2,444	11.65	3.89	9.94	26.19	0.60	10.52	1.01	0.60	12.72	1.01	**0.59**	15.59	1.01
ED-11-72	2,447	4	2,446	3.96	4.03	9.50	27.50	0.82	11.29	1.01	**0.81**	14.75	1.01	**0.81**	18.32	1.01
ED-11-63	2,510	4	2,509	14.75	4.22	11.09	27.08	0.67	12.39	1.01	0.66	15.83	1.01	**0.65**	20.58	1.01
ED-11-54	2,512	4	2,511	5.23	4.28	10.37	27.32	0.67	12.85	1.01	**0.66**	16.77	1.01	**0.66**	21.93	1.01
ED-11-57	2,559	4	2,559	16.09	4.42	10.28	28.71	0.77	12.72	1.01	**0.75**	16.95	1.01	**0.75**	22.46	1.01
ED-11-76	2,581	4	2,581	3.77	3.81	17.86	30.16	0.11	11.17	1.00	0.11	14.18	1.00	**0.10**	16.30	1.00
ED-11-42	2,598	4	2,598	2.11	4.10	21.40	30.32	0.16	11.66	1.00	0.16	32.91	1.00	**0.15**	17.24	1.00
ED-11-56	2,632	4	2,630	3.69	4.60	11.46	30.21	0.56	12.94	1.01	**0.55**	15.88	1.01	**0.55**	19.63	1.01
ED-11-61	2,726	4	2,726	6.46	4.88	6.72	32.41	0.96	14.31	1.01	0.96	17.88	1.01	**0.95**	22.94	1.02
ED-11-47	2,819	4	2,819	4.16	4.72	19.94	34.68	0.24	12.28	1.00	**0.23**	14.20	1.00	**0.23**	16.66	1.01
Average				11.52	1.73	10.58	≥ 18.32	1.12	5.79	1.02	1.10	7.45	1.02	1.09	10.14	1.02

Table 3.3: Performance Metrics of the Selected Approximation Algorithms for Solving SOC Instances with More than 100 Items

Instance	n	m	$(X_l^f)^{max}$	KwikSort		Deterministic KwikSort		LPKwikSort		BestInput		Spearman's footrule		Approximate Condorcet Partitioning								
														h=30			h=40			h=50		
				Gap %	Time	Gap %	Time	Gap %	Time	Gap %	Time	Gap %	Time	Gap %	Time	AF	Gap %	Time	AF	Gap %	Time	AF
ED-15-12	100	4	99	11.80	0.06	11.28	0.45	0.42	6.25	5.49	0.02	6.14	0.06	1.73	0.59	1.06	1.20	1.23	1.06	1.37	1.87	1.06
ED-15-42	100	4	100	12.62	0.06	9.45	0.44	0.85	6.16	13.0	0.02	6.25	0.05	1.38	0.64	1.04	0.94	1.31	1.04	1.24	1.84	1.04
ED-15-28	102	4	99	10.37	0.06	4.87	0.47	0.15	6.50	1.96	0.02	7.97	0.06	1.61	0.64	1.07	1.11	1.38	1.07	0.76	1.94	1.07
ED-15-36	102	4	100	14.30	0.06	7.26	0.48	0	6.41	0.20	0.02	7.04	0.06	0.93	0.61	1.08	1.22	1.33	1.08	0.69	2.00	1.08
ED-15-05	103	4	94	12.44	0.06	11.94	0.55	0.11	10.24	10.92	0.03	6.25	0.06	0.21	0.95	1.03	0.11	1.05	1.03	0.21	1.64	1.03
ED-11-03	103	5	90	6.48	0.06	12.79	0.47	2.77	6.56	9.50	0.03	7.09	0.06	2.85	1.11	1.05	2.83	1.05	1.05	2.85	1.53	1.05
ED-15-29	106	4	105	13.97	0.06	9.46	0.59	0.32	7.39	1.49	0.02	5.52	0.06	1.28	0.66	1.06	1.44	1.41	1.06	1.28	1.85	1.06
ED-15-07	110	4	106	13.90	0.06	13.23	0.59	0.16	7.86	2.40	0.02	6.29	0.08	1.14	1.06	1.05	0.73	1.13	1.05	0.57	2.00	1.05
ED-15-22	112	4	110	10.21	0.08	16.16	0.67	0.05	8.67	0.90	0.02	5.82	0.06	1.30	0.66	1.07	1.30	1.55	1.07	1.21	1.92	1.07
ED-15-18	115	4	112	10.78	0.08	21.88	0.70	0	9.28	1.32	0.03	6.13	0.08	1.37	0.73	1.06	0.75	1.61	1.06	0.58	2.02	1.06
ED-15-25	115	4	114	10.12	0.06	15.24	0.70	0.08	9.24	1.44	0.03	7.25	0.08	1.79	0.72	1.06	0.92	1.64	1.06	1.00	1.98	1.06
ED-15-09	115	4	115	14.48	0.06	15.85	0.69	0.22	9.44	1.27	0.03	5.01	0.08	1.48	1.19	1.06	1.53	1.35	1.06	0.92	2.3	1.06
ED-15-20	122	4	116	16.08	0.06	24.35	0.94	0.19	10.99	1.38	0.03	6.97	0.09	3.06	0.78	1.08	1.91	1.87	1.08	1.48	1.91	1.08
ED-15-17	127	4	124	12.47	0.08	9.47	1.00	0.10	12.72	0.37	0.03	6.43	0.10	1.99	0.86	1.07	1.66	2.03	1.07	1.27	2.00	1.07
ED-15-33	128	4	126	13.69	0.08	15.34	1.00	0.70	12.5	2.46	0.03	6.74	0.08	3.47	0.80	1.08	2.75	2.02	1.08	1.94	2.13	1.08
ED-15-40	131	4	131	15.12	0.09	15.65	1.14	0.26	13.52	0.96	0.03	8.33	0.09	1.80	0.78	1.06	1.63	1.30	1.06	1.63	2.83	1.06
ED-15-23	142	4	135	21.48	0.09	15.33	1.55	0.03	17.58	1.04	0.05	7.50	0.11	2.19	0.91	1.07	1.93	1.37	1.07	1.77	3.06	1.07
ED-15-32	153	4	153	15.18	0.13	17.97	2.09	0.26	21.56	1.04	0.05	7.93	0.13	2.57	0.94	1.07	2.23	1.59	1.07	1.44	3.67	1.07
ED-15-14	163	4	160	14.50	0.16	18.28	2.67	0.04	26.32	0.65	0.05	8.91	0.14	2.00	1.09	1.07	1.59	3.25	1.07	1.76	2.83	1.07
ED-15-01	240	4	240	3.37	0.28	10.59	12.13	0.35	84.28	8.40	0.13	4.25	0.33	0.89	1.67	1.03	0.83	2.77	1.03	0.83	7.77	1.03
ED-11-01	240	5	229	4.38	0.39	8.82	12.38	2.22	85.31	7.09	0.19	5.22	0.39	2.37	1.75	1.04	2.27	2.58	1.04	2.28	3.89	1.04
ED-15-03	242	4	242	4.87	0.30	10.03	13.03	1.08	95.98	8.39	0.11	5.59	0.33	3.69	1.75	1.10	3.08	2.80	1.12	3.09	4.44	1.12
ED-11-02	242	5	239	6.22	0.47	10.75	12.20	3.89	93.31	7.37	0.19	6.94	0.39	3.91	2.02	1.06	3.72	6.86	1.06	3.83	5.22	1.06
Average				11.69	0.10	13.30	1.26	0.62	14.66	3.87	0.04	6.59	0.10	1.96	0.93	1.06	1.64	1.72	1.06	1.48	2.35	1.06

3.3 Lower Bounding Techniques

This section focuses on the lower bounds on KEMENY-AGG. As mentioned earlier, obtaining high quality is crucial for evaluating the solution quality of heuristic methods; furthermore, the instance-specific approximation ratio of ACP requires a lower bound on KEMENY-AGG. This chapter generalizes some of the existing lower bounds for strict rankings to the case of non-strict rankings and proposes shortcuts for reducing the run time of these techniques. More specifically, we use Condorcet

criterion variations and the Constraint Relaxation method to accelerate the lower bounding process.

The rest of the section is organized as follows. Section 3.3.1 reviews the lower bound obtained from pairwise comparison information for strict rankings and generalizes this method for non-strict rankings. Section 3.3.2 focuses on cycle-based lower bounding techniques and leverages social choice properties that KEMENY-AGG is guaranteed to satisfy for the purpose of accelerating cycle-based methods. Section 3.3.3 focuses on the linear programming (LP) relaxation lower bounding techniques. Section 3.3.4 describes the experiments and discusses the results.

3.3.1 Pairwise Comparison Methods

Davenport and Kalagnanam (2004) proposed the first lower bound for KEMENY-AGG with strict ranking as follows

$$LB_0 = 2 \sum_{(i,j)\in\mathbf{\Lambda}} \min(s_{ij},\, s_{ji}). \tag{3.8}$$

Eq. (3.8) obtains a lower bound on $d_{KS}(\overline{\boldsymbol{\sigma}})$ by simply summing the smallest contribution of all distinct pair of items. Note that Eq. (3.8) has been scaled by a factor of 2 herein to facilitate the generalization of this lower bound for non-strict rankings.

Proposition 3 *Given an instance of KEMENY-AGG, a lower bound on $d_{KS}(\overline{\boldsymbol{\sigma}})$ is given by*

$$LB_1 = \sum_{(i,j)\in\mathbf{\Lambda}} \min(2s_{ij} + t_{ij},\, 2s_{ji} + t_{ij},\, s_{ij} + s_{ji}). \tag{3.9}$$

Similar to LB_0, Eq. (3.9) obtains a lower bound on $d_{KS}(\overline{\boldsymbol{\sigma}})$ by simply summing the smallest contribution of all distinct pair of items. LB_1 reduces to LB_0 when the input

rankings are strict, i.e., if $t_{ij} = 0 \; \forall i, j \in \boldsymbol{\mathcal{X}}$.

3.3.2 Cycle-Based Methods

Cycle-Based Methods for Strict Rankings

KEMENY-AGG with strict rankings can be solved via the Weighted Minimum Feedback Arc Set Problem (WMFASP), and vice versa (Kenyon-Mathieu and Schudy, 2007; Conitzer *et al.*, 2006). Let $G = (V, E)$ be a weighted directed graph where V is the set of vertices and E is the set of arcs (edges). The objective of WMFASP is to find a subset of arcs $E' \subset E$ with minimum weight such that its removal would make the resulting graph, i.e., $G' = (V, E \backslash E')$, acyclic (Younger, 1963). Conitzer *et al.* (2006) provided various lower bounds for the equivalent WMFASP of KEMENY-AGG with strict rankings on the *pairwise majority graph*. The nodes of this graph are the items; there is a directed arc from i to j if $p_{ij} > p_{ji}$ with a weight of $w_{ij} = p_{ij} - p_{ji}$; and there is no arc from i to j and vice versa if $p_{ij} = p_{ji}$. The lower bounds on the WMFASP pairwise majority graph in Conitzer *et al.* (2006) do not provide any information on how to obtain the respective lower bounds on the equivalent KEMENY-AGG. For this reason, Milosz and Hamel (2018) utilized methods developed in Conitzer *et al.* (2006) to improve LB_0.

Recall that LB_0 considers the smallest contribution for each pair of items. However, the resulting solution obtained by this selection may not be transitive as it may contain preference-cycles, which can make this bound unattainable. Nevertheless, this information can be utilized to improve this lower bound.

Let $\overline{C} = \{c_1, c_2, \ldots, c_s\}$ be any set of edge-disjoint preference-cycles. For each cycle, the consensus ranking disagrees with at least one of the edges in the cycle (Conitzer *et al.*, 2006). Hence, the lower bound can be improved by adding the

minimum cost of reverting an edge of cycle, i.e., the minimum cost of breaking the preference-cycle. Therefore, a new lower bound can be calculated as follows (Conitzer *et al.*, 2006; Milosz and Hamel, 2018):

$$LB_2 = LB_0 + \sum_{c_r \in \overline{C}} \min_{(i,j) \in c_r} w_{ij}. \tag{3.10}$$

When only edge-disjoint preference-cycles are considered, a part of the cycles remains unused. Conitzer *et al.* (2006) proposed a method to leverage this underutilized information. Let $C = \{c_1, c_2, \ldots, c_s\}$ be any set of preference-cycles, and $\delta\left((i,j), c_r\right)$ be an indicator function which is set to 1 if $(i,j) \in c_r$, and 0 otherwise. Additionally, let $v_l = \min_{(i,j) \in c_r} \{w_{ij} - \sum_{q=1}^{l-1} \delta\left((i,j), c_q\right) . v_q\}$ (Conitzer *et al.*, 2006). Intuitively, v_l calculates the minimum portion of the weights of c_r edges that have not been used by prior cycles in C. A lower bound on KEMENY-AGG with strict rankings can be calculated as Conitzer *et al.* (2006); Milosz and Hamel (2018)

$$LB_3 = LB_0 + \sum_{c_r \in C} v_l. \tag{3.11}$$

Notice that LB_3 is at least as good as LB_2 (Conitzer *et al.*, 2006).

Cycle-Based Methods for Non-Strict Rankings

LB_1 provides a lower bound on KEMENY-AGG with non-strict rankings using pairwise comparison information by considering the smallest among all three possible values. Similar to the case of strict rankings, the resulting ranking of this selection may contain preference-cycles, which can be similarly broken to boost LB_1.

Similar to Conitzer *et al.* (2006) and Milosz and Hamel (2018), we focus only on preference-cycles of length 3 for the purposes of simplicity and computational

efficiency. These preference-cycles are much easier to find, and every preference-cycle of length 4 or higher contains at least a preference-cycle of length 3 (Gass, 1998).

As mentioned earlier, KEMENY-AGG with strict rankings and WMFASP are equivalent problems. However, this claim has not yet been proven for non-strict rankings. We reckon that, to the best of our knowledge, it may not be possible to represent KEMENY-AGG with non-strict rankings via an equivalent WMFASP. The reason is that, for every pair of distinct items (i, j), there are 3 parameters involved, namely p_{ij}, p_{ji}, t_{ij}. Additionally, in the case of strict rankings, items i, j, k form a preference-cycle if $i \succ j \succ k \succ i$; however, in the case of non-strict rankings, there are additional types of preference cycles as shown in Fig. 3.1, where arc (i, j) is drawn if $i \succ j$; and arcs (i, j) and (j, i) are simultaneously drawn if $i \approx j$. In the case of strict rankings, it is possible to break a preference-cycle of length 3 by reversing the edge with the lowest weight, however, this method cannot be applied to non-strict rankings, as edges are not weighted; additionally, reversing certain individual edges may not break the cycle. For example, reversing edge (k, j) in Fig. 3.1 (b) does not make the resulting graph acyclic. Consequently, it is not possible to directly apply the previously reviewed techniques for strict rankings to the case of non-strict rankings. We propose a new method to boost LB_1 using preference-cycles. Let $\overline{C} = \{c_1, \ldots, c_s\}$

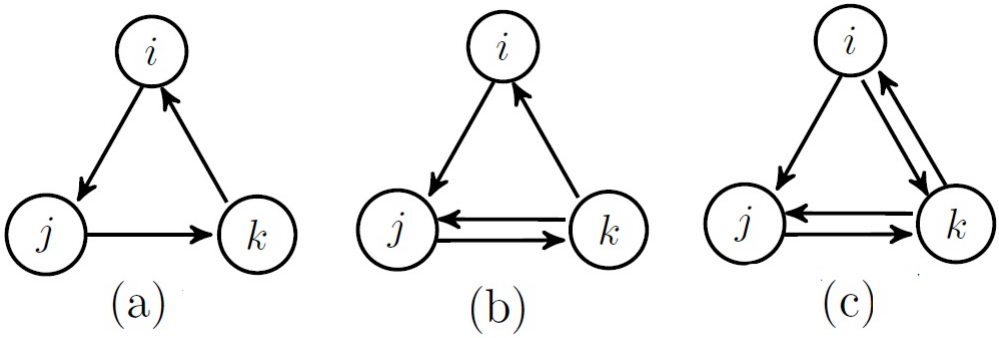

Figure 3.1: Preference-cycles in Non-strict Rankings (Yoo and Escobedo, 2021)

be any set of edge-disjoint preference-cycles of length 3. For each cycle $c_r \in \overline{C}$, we explicitly evaluate all 13 possible non-strict rankings (i.e., acyclic preferences) of 3 items, and we define X_r^* as the minimum d_{KS} distance of KEMENY-AGG restricted to the items in cycle c_r. X_r^* is the minimum possible contribution of items i, j, k in d_{KS}^*, i.e., consensus ranking. On the other hand, the contribution of pairs $(i, j), (i, k)$, and (j, k) to LB_1, denoted by d_{ijk}, is equal to

$$d_{ijk} = 2\left[\min d_{KS}(\sigma_{ij}) + \min d_{KS}(\sigma_{ik}) + \min d_{KS}(\sigma_{jk})\right]. \tag{3.12}$$

We remark that Eq. (3.12) has been multiplied by 2 since d_{KS} counts each pair of items twice.

Hence, the improvement caused by breaking cycle c_r, denoted by Q_r, is equal to

$$Q_r = X_r^* - d_{ijk}.$$

As a result, an improved lower bound can be obtained as

$$LB_4 = LB_1 + \sum_{c_r \in \overline{C}} Q_r. \tag{3.13}$$

Given a set of cycles, it is possible to construct different edge-disjoint sets that can result in different values of LB_4. Since LB_1 is a fixed value, LB_4 is maximized by focusing on the second term of Eq. (3.13).

The problem of finding the set of disjoint cycles that yields the highest LB_4 value can be formulated as a weighted node packing problem (WNP). Let $G = (V, E, W)$ be an undirected graph where V is the set of nodes, E is the set of edges, and W is the set of nodes' weights. The goal of WNP is to find a subset of nodes with maximum total weight such that no pair of nodes share an edge (Nemhauser and Sigismondi,

1992). Here, V is the set of cycles, i.e., $V = C$. There is an edge between cycles $c_r, c_g \in C$ if c_r and c_g are not edge-disjoint. The weight of node c_r is its improvement to the lower bound, i.e., Q_r.

Let $C = \{c_1, \ldots, c_s\}$ be the set of all cycles of length 3, which may not necessary be edge-disjoint. Given C, the ensuing optimization problem maximizes LB_4. Beforehand, we introduce the decision variables and parameters of the model. Let decision variable v_l be equal to 1 if cycle $c_r \in C$ is in the selected set of edge-disjoint cycles, and 0 otherwise. Additionally, let Ξ be the set of cycle pairs that share an edge. The binary programming formulation is given by

$$\max \quad \sum_{c_r \in C} Q_r v_r \tag{3.14a}$$

$$\text{subject to} \quad v_r + v_g \leq 1, \quad \forall (c_r, c_g) \in \Xi \tag{3.14b}$$

$$v_r \in \{0, 1\}, \quad \forall c_r \in C. \tag{3.14c}$$

Objective function (3.14a) maximizes LB_4; Constraint (3.14b) enforces that, whenever cycles c_r and c_g share one edge, at most one of them can belong to the set of edge-disjoint cycles; and Constraint (3.14c) specifies the domain of the decision variables. WNP is an NP-hard problem for general graphs and even finding an approximation algorithm for this problem is NP-hard (Nemhauser and Sigismondi, 1992). WNP has stronger formulations using cliques (Nemhauser and Sigismondi, 1992). However, solving this problem to optimality still may be computationally demanding. Here, we propose a simple add-swap heuristic method to find a high quality set of edge-disjoint cycles for our problem of interest. The pseudocode of the proposed method is presented in Algorithm 3, which begins by sorting cycles based on non-increasing improvement in the lower bound, i.e., Q-values. Additionally, let C' be the working

Algorithm 3: Lower Bound Improvement Using Preference-cycles

Input : Set of cycles of length 3 (C), $Q = [Q_r] \in \mathbb{Z}^{|C|}$
Output: Overall improvement in the lower bound

Sort cycles based on non-increasing Q-values;
Discard cycles with zero improvement;
$C' \leftarrow \emptyset$; `// Set of edge disjoint cycles`
$\overline{Q} \leftarrow 0$; `// Overall improvement in the lower bound`
for $l = 1$ **to** $|C|$ **do**
    if *Cycle* c_r *is mutually edge-disjoint with all cycles in* C' **then**
        $C' \leftarrow C' \cup c_r$;
        $\overline{Q} \leftarrow \overline{Q} + Q_r$
while *True* **do**
    Swap cycles $c_m, c_n \notin C'$ with $c_g \in C'$ if this swap allows the set to remain edge-disjoint and increases $\overline{Q}$;
    Otherwise, set to False
return $\overline{Q}$

set of edge-disjoint cycles, which initially is set to be empty. Next, starting with the cycle that yields the highest improvement, the algorithm adds this cycle to C' if its addition keeps the working set edge-disjoint. After this step, the algorithm checks whether it is possible to swap one cycle in C' with two mutually edge-disjoint cycles that are not in C', such that this swap keeps C' to remain edge-disjoint and simultaneously increases the overall lower bound improvement of C'.

Scaling up Cycle-Based Methods with Social Choice Properties

Detecting preference-cycles of length 3 has a time complexity of $O(n^3)$, which makes this lower bound boosting technique suitable only for small- to medium-sized problems. Here, we use partitioning to reduce the run time of this process. Since applying XCC and NXCC has a time complexity of $O(n^2)$ (Yoo and Escobedo, 2021; Truchon, 1998), this modification can make cycle detection operation less expensive and maybe even suitable for certain large-scale problems. Let $\boldsymbol{X} = \{X_1, X_2, \cdots, X_w\}$ be the par-

titioned problem according to NXCC, denoted by NXCC partition. In the original problem, all $\binom{n}{3}$ triplet items must be checked for possible preference cycles, however, in the partitioned problem with NXCC it is sufficient to evaluate all subsets of $\boldsymbol{X}$ for possible preference cycles independently as there are no preference cycles between items from different subsets. Hence, in the partitioned problem only $\binom{|X_1|}{3} + \binom{|X_2|}{3} + \ldots + \binom{|X_w|}{3}$ triplets must be checked.

Proposition 4 *Given an instance of* KEMENY-AGG, *the set of cycles obtained from the partitioned problem by XCC and NXCC and the original problem are the same.*

All the items that form a preference-cycle must belong to the same subset in the NXCC partition Truchon (1998). Therefore, there are no preference cycles between items from different subsets.

3.3.3 LP-relaxation-Based Methods

It is well known that the LP-relaxation version of a minimization Binary Linear Programming (BLP) model provides a valid lower bound on the respective BLP model. Conitzer *et al.* (2006) explored this type of lower bound on the WMFASP version of KEMENY-AGG with strict rankings. This method yields tighter bounds than cycle-based methods; however, it takes more time as well. This is not surprising, since the exact formulation of KEMENY-AGG, and hence the LP-relaxation version, has $O(n^3)$ constraints, which can be cumbersome to solve for large values of n. However, a large portion of these constraints are trivially satisfied at the optimal solution (Pedings *et al.*, 2012), which is a fact that can be utilized to simplify the solution to the LP-relaxation models of KEMENY-AGG. Here, we explore an item exact solution approach for solving the LP-relaxation problem, the Constraint Relaxation (CR) method (Dantzig *et al.*, 1954, 1959). CR is an iterative optimization

approach that begins with a relaxed version of a problem's exact formulation; the relaxation usually excludes a large number of the constraints and is, hence, easier to solve. Here, the cycle-prevention constraints are excluded from the model at first. At each iteration, the working relaxation is solved to optimality and the solution is analyzed to determine if any of the excluded constraints are violated, in which case the respective constraints are added back into the working relaxation model. This process is repeated until there are no such violations.

Furthermore, the process of obtaining a lower bound using the LP-relaxation method can be accelerated by deploying NXCC partitioning. If the instance of interest is partitionable, the original problem can be equivalently solved as a collection of smaller subproblems, which can decrease run times.

3.3.4 Computational Results

This section compares the quality and run time of the various lower bounding techniques for non-strict rankings discussed in this chapter. Condorcet-based partitioning was performed using the algorithm proposed in Yoo and Escobedo (2021), which works by carrying out sequential pairwise comparisons. All experiments were carried out on a computer with an Intel(R) Xeon(R) CPU E5-2680 @ 2.40 GHz with 64 GB RAM. The optimization models were solved using CPLEX solver version 12.10.0.

We use the Mallows model (see Section 2.1) to generate synthetic instances. Using the Mallows model, we can control the difficulty of generated instances and investigate the performance of the algorithms under different conditions.

Doignon *et al.* (2004) introduced the Repeated Insertion Model (RIM) for generating strict rankings, which encompasses the Mallows model as a special case. To describe RIM, assume without loss of generality, that the ground truth ranking $\boldsymbol{\sigma}'$ is

Table 3.4: Computational Results of Different Lower Bounding Techniques

N=50															
ϕ	Pairwise Comparison (PC)				Cycle-Based (CB)					LP-relaxation (LPR)					
	LB			Time	LB			Time		LB			Time		
	Ave	Min	Max	PC	Ave	Min	Max	CB	CB + NXCC	Ave	Min	Max	LPR	LPR + CR	LPR + CR + NXCC
0.8	6490.25	5974	6892	0.00	6510.35	6004	6906	0.06	0.01	6513.05	6006	6906	4.20	1.68	0.22
0.85	8651.65	7940	9418	0.00	8684.85	7956	9452	0.06	0.03	8689.7	7956	9458	4.27	1.84	0.41
0.9	11782.75	10726	12854	0.00	11842.35	10796	11806.5	0.09	0.10	11852.55	10810	12940	4.24	1.94	1.20
0.95	16705.15	15633	17906	0.00	16856.75	15777	16790.5	0.37	0.39	16892.55	15811	16817.5	4.35	1.93	1.86

N=100															
ϕ	Pairwise Comparison (PC)				Cycle-Based (CB)					LP-relaxation (LPR)					
	LB			Time	LB			Time		LB			Time		
	Ave	Min	Max	PC	Ave	Min	Max	CB	CB + NXCC	Ave	Min	Max	LPR	LPR + CR	LPR + CR + NXCC
0.8	14263.8	13585	15073	0.01	14304.8	13615	15109	0.38	0.03	14311.7	13617	15117	35.8	15.20	1.34
0.85	19654.55	18644	20615	0.01	19725.55	18724	20701	0.39	0.08	19738.95	18738	20727	35.63	15.57	1.62
0.9	29167.35	27445	30625	0.01	29308.45	27583	30799	0.53	0.51	29335.85	27623	30831	35.56	15.98	10.01
0.95	49735.15	46784	52936	0.01	50059.35	47030	53310	2.16	2.19	50137.35	47091	53392	36.11	15.38	15.41

N=150															
ϕ	Pairwise Comparison (PC)				Cycle-Based (CB)					LP-relaxation (LPR)					
	LB			Time	LB			Time		LB			Time		
	Ave	Min	Max	PC	Ave	Min	Max	CB	CB + NXCC	Ave	Min	Max	LPR	LPR + CR	LPR + CR + NXCC
0.8	22276.6	21578	23189	0.02	22345.3	21642	23273	1.26	0.08	22357.1	21652	23291	122.32	52.67	3.43
0.85	30640.1	29563	31850	0.02	30745.6	29669	31974	1.26	0.18	30766.75	29692	32004	122.30	56.02	3.96
0.9	47121.75	44486	49368	0.02	47324.25	44686	49624	1.49	1.46	47364.05	44718	49682	121.75	55.65	31.92
0.95	85673.6	84089	86765	0.02	86168	84577	87315	6.75	6.76	86282	84713	87451	121.06	58.80	58.81

N=200															
ϕ	Pairwise Comparison (PC)				Cycle-Based (CB)					LP-relaxation (LPR)					
	LB			Time	LB			Time		LB			Time		
	Ave	Min	Max	PC	Ave	Min	Max	CB	CB + NXCC	Ave	Min	Max	LPR	LPR + CR	LPR + CR + NXCC
0.8	29850.1	29046	30894	0.03	29935.1	29118	30980	2.79	0.11	29948.1	29120	31002	294.98	131.80	8.28
0.85	41825.2	40709	43203	0.03	41969.4	40855	43339	2.86	0.28	41992	40879	43363	296.57	135.50	6.71
0.9	64462.8	62991	68004	0.04	64742.6	63239	68290	3.40	2.84	64792.7	63285	68332	298.87	128.01	61.59
0.95	122121.6	118772	125625	0.03	122824.4	119506	126353	13.68	13.72	122968.6	119645	126533	292.13	130.89	130.92

the permutation $(1, 2, \ldots, n)$. The method starts by placing item 1 into an initially empty working ranking vector; in each succeeding iteration and until the target size is reached, the next item from $\boldsymbol{\sigma}'$ is inserted in a specific position in the working ranking vector based on the Mallows probabilities. Specifically, item i is inserted before item $j < i$ in the working ranking vector with probability $p_{ij} = \phi^{i-j}/(1 + \phi + \cdots + \phi^{i-1})$. Yoo *et al.* (2020) developed a modified RIM sampling process for generating non-strict rankings, which is used herein. In this sampling process, after generating strict rankings via RIM, a random number u is drawn from a uniform distribution $U(1, n-1)$, and the item with rank u is tied with the item with the next higher (i.e., worse) rank. The process is repeated until the number of items that are tied reaches a specific threshold, herein set to $0.25n$. Please refer to Doignon *et al.* (2004) and Yoo *et al.* (2020) for more information.

The tested parameter settings are $\phi \in \{0.8, 0.85, 0.9, 0.95\}$, $n \in \{50, 100, 150, 200\}$, and $m = 20$; we chose only high values of ϕ because they are more difficult to solve, as they correspond to low group cohesion and higher noise levels (Yoo *et al.*, 2020). For each combination of (ϕ, n), we perform 20 replications. Since the ground truth ranking used in the Mallows model are the same for each combination of (ϕ, n), the d^*_{KS} values very close to each other. For all three tested lower bounding techniques, the experimental results shown in Table 3.4 report the average, minimum, and maximum lower bound over the 20 replications for each combination of (ϕ, n). Furthermore, Table 3.4 reports the geometric mean run time of the pairwise comparison (PC); the run time of the cycle-based method (CB) and its run time with NXCC (CB + NXCC); and the run time of LP-relaxation (LPR), its run time with CR (LPR + CR), and its run time with CR and NXCC (LPR + CR + NXCC) over the 20 replications for each combination of (ϕ, n).

Instances with a lower ϕ value yielded a partition with more subsets than those with a higher value. This was expected since higher values of ϕ correspond to more noise in the generated rankings, which induces less agreement on the relative ordering of items in these instances. All tested instances with a ϕ value of 0.8, 0.85, 0.9 yielded a non-trivial NXCC partition, however, all tested instances with a ϕ value of 0.95 were not partitionable. Whenever NXCC yielded a non-trivial partition, it was able to accelerate the preference-cycle detection process rather significantly, especially for instances with a ϕ value of 0.8 and 0.85 where the NXCC partition had the most subsets.

For all tested combinations of (ϕ, n), the LP-relaxation method achieves a better average, minimum, and maximum lower bound than the other two techniques; furthermore, this was true for all the individual instances. On the other hand, LP-relaxation had the highest average run time for all tested combinations of (ϕ, n);

this was true for all the individual instances as well. As expected, the cycled-based technique achieves a better lower bound than the pairwise comparison method; on the other hand, it takes significantly more time. The pairwise comparison method achieves the worst bounds. However, the highest run time of this technique was only 0.04 seconds, which makes this method very attractive whenever a very fast lower bounding technique is required.

The cycle-based technique was able to improve the pairwise comparison lower bound in up to a handful of seconds. In fact, this technique was able to achieve bounds that were competitive with the LP-relaxation technique in far less time. Deploying NXCC makes this method even more attractive as, whenever NXCC yielded a non-trivial partition, it was able to accelerate the cycle-based technique rather impressively, especially for instances with a ϕ value of 0.8 and 0.85. For example, NXCC was able to reduce the average run time of this technique from 2.79 to 0.11 seconds, which represents a 25x computational speedup.

Even though theLP-relaxation technique achieves the best bounds, its run time is considerably larger than the other two methods, which may make this technique somewhat less useful in real-world applications. However, incorporating the B&C method was able to reduce the run time significantly, specifically by more than half in all tested instances. Moreover, NXCC was able to accelerate the LP-relaxation technique with B&C rather remarkably for instances with a ϕ value of 0.8,0.85, and 0.9. For example, it was able to reduce the average run time of this technique from 135.5 to 6.71 seconds, which represents a 20x computational speedup.

It is important to remark that the maximum NXCC partitioning time over all tested instances was only 0.09 seconds, which makes this computationally inexpensive operation worthwhile to lower bounding techniques for KEMENY-AGG.

All in all, the pairwise comparison method is suitable when a very fast lower

bounding technique is required. When high-quality bounds are desired, the cycle-based method combined with NXCC partitioning is probably the best candidate, as it can produce competitive bounds in up to a handful of seconds. Finally, LP-relaxation combined with B&C and NXCC method produces the tightest bounds in considerably more time.

3.4 Concluding Remarks

This chapter explores the partitioning of KEMENY-AGG based on Condorcet extensions. These partitioning schemes offer theoretical guarantees that enable the decomposition of certain large instances of this NP-hard problem into a set of smaller subproblems that can be solved independently. Since there may exist more than one partition that satisfies the criteria of the Condorcet variants, we formalize the concept of the finest-Condorcet partition, which is designed to provide the highest possible computational advantages among all such partitions. We specify the requirements of the finest-Condorcet partition, prove its uniqueness, and derive an algorithm for its construction. Condorcet Partitioning is useful for a small portion of problem instances, as it often yields a few large subsets which may be too difficult to solve with exact methods. To overcome this issue, we propose Approximate Condorcet Partitioning (ACP) that breaks down these larger subsets based on the number of times an item is pairwise preferred over other items. The resulting partition has more subsets than the finest-Condorcet partition and is therefore easier to solve. Furthermore, we propose an efficient solution technique for strict and non-strict rankings, which is accompanied by instance-specific approximation factors. Although the approximation factors are not constant, ACP often achieves better solution guarantees than all known approximation factors, including all instances tested herein. The average approximation ratio for the strict and non-strict rankings instances tested herein was

1.06 and 1.03, respectively, whereas the best-known approximation factors for strict and non-strict rankings are 4/3 and 9, respectively. Experiments on a variety of very large benchmark instances demonstrate the scalability and robustness of the proposed approximation algorithm. The conducted experiments on real-world instances showed that LPKiwkSort and the proposed solution technique via ACP had the best and second best performances in terms of solution quality on strict rankings, differing by no more than 2.8 percentage points; however, ACP was on average nearly 5 times faster than LPKiwkSort. On the other hand, ACP had a dominant performance on non-strict rankings, achieving near-optimal solutions on the majority of the tested instances.

Additionally, this chapter explores the lower bounding techniques for KEMENY-AGG with non-strict rankings. In particular, it generalizes the existing techniques, i.e., pairwise comparison, cycle-based, and LP-relaxation, for strict rankings to the case of non-strict rankings. Additionally, it utilizes partitioning using variations of the formative Condorcet criterion and Constraint Relaxation (CR) methods to accelerate the lower bounding process. The experimental results demonstrate that LP-relaxation provides the tightest bounds, but it is substantially computationally more demanding than other techniques. However, deploying partitioning and CR can drastically reduce the run time of this technique. Moreover, cycle-based method produces high-quality bounds in a reasonable time, additionally, the run time of this method can further be reduced by partitioning.

Chapter 4

THE GENERALIZED RANK AGGREGATION PROBLEM

4.1 Overview

This chapter delves into the rank aggregation problem under the generalized Kendall-tau distance (Fagin *et al.*, 2004), which contains KEMENY-AGG as a special case. First, it derives exact and heuristic solution methods for the generalized problem. Second, it introduces a social choice property that encloses existing variations of the Condorcet criterion as special cases, thereby expanding this seminal social choice concept beyond Kemeny aggregation for the first time. Through a specialized partitioning algorithm designed herein to implement this property, many instances of the NP-hard general rank aggregation problem can be decomposed into smaller subproblems, while guaranteeing that solving them independently still produces the optimal solution to the original problem. Experiments on two benchmark datasets conducted herein show that the featured exact and heuristic solution methods can benefit from this property. Finally, this work derives new theoretical insights into the effects of the generalized Kendall-tau distance penalty parameter on the optimal ranking and on the proposed social choice property.

The rest of this chapter is organized as follows. Section 4.2 introduces notations used throughout the paper and establishes some preliminaries. Section 4.3 introduces various exact and heuristic methods. Section 4.4 generalizes the Condorcet criterion and its variants. Section 4.5 studies the effect of the generalized Kendall-tau distance penalty parameter on the optimal solution. Section 4.6 presents the computational results. Finally, Section 4.7 concludes the chapter.

4.2 Notation and Preliminaries

Fagin *et al.* (2004) proposed a generalization of the Kendall-tau distance for non-strict rankings using bucket orders, otherwise known as weak orders. A bucket order $\boldsymbol{\mathcal{B}}$ is a transitive, total, and reflexive binary relation $\succ$ in which buckets $B_1, \ldots, B_t$ form a partition of $\boldsymbol{\mathcal{X}}$ such that $i \succ j$ if and only if $i \in B_k$ and $j \in B_{k'}$, with $k < k'$. Members of the same bucket are considered as being tied. The position of bucket B_k is defined as $pos(B_k) = (\sum_{k'<k} |B_{k'}|) + (|B_k| + 1)/2$, and it indicates the average location within bucket B_k. A bucket order becomes a linear order when the cardinality of all buckets equals one. A non-strict ranking $\boldsymbol{\sigma}$ can be mapped to a bucket order by letting $\sigma_i = pos(\overline{B})$, where $\overline{B}$ is the bucket containing item i (Fagin *et al.*, 2004).

Next, we restate the definition of the generalized Kendall-tau distance introduced by Fagin *et al.* (2004). Given a fixed penalty parameter $0 \leq p \leq 1$ and two rankings $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$, let $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ be the contribution to the distance function, for each pair $(i, j) \in \boldsymbol{\Lambda}$. There are three cases with respect to the relative orderings of items i and j in $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$:

Case 1. There is a strict ordering between i and j in $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$. If i and j are in the same order in both rankings, set $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 0$; otherwise, set $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 1$.

Case 2. Both rankings tie i and j. In this case, set $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 0$.

Case 3. One of the rankings ties i and j, but not the other. In this case, set $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = p$.

Piecing together the above three cases, $K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2)$ can be succinctly written as

$$K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \begin{cases} 1 & \text{if } \; (i \succ_{\boldsymbol{\sigma}^1} j \; \wedge \; j \succ_{\boldsymbol{\sigma}^2} i) \; \vee \; (j \succ_{\boldsymbol{\sigma}^1} i \; \wedge \; i \succ_{\boldsymbol{\sigma}^2} j) \\ p & \text{if } \; (i \approx_{\boldsymbol{\sigma}^1} j \wedge (i \succ_{\boldsymbol{\sigma}^2} j \; \vee \; j \succ_{\boldsymbol{\sigma}^2} i)) \vee (i \approx_{\boldsymbol{\sigma}^2} j \wedge (i \succ_{\boldsymbol{\sigma}^1} j \; \vee \; j \succ_{\boldsymbol{\sigma}^1} i)) \\ 0 & \text{otherwise.} \end{cases}$$

Taking this one step further, the *Kendall-tau distance with penalty parameter* p, denoted as $K^{(p)}$, can be abbreviated as

$$K^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = \sum_{(i,j)\in\boldsymbol{\Lambda}} K_{ij}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2). \tag{4.1}$$

Note that Case 1 corresponds to a full rank reversal, and Case 3 corresponds to a partial rank reversal. Additionally, $K^{(p)}$ induces the Kemeny-Snell distance (scaled by 1/2) and the Kendall-tau distance as special cases, namely for $p = 1/2$ and $p = 0$, respectively. The $K^{(p)}$ distance is a metric for $1/2 \leq p \leq 1$, a near metric for $0 < p < 1/2$, and not a metric for $p = 0$ (Fagin *et al.*, 2004); for the rest of the paper, we focus on the values of p for which $K^{(p)}$ is a metric. We close this section with an example that helps illustrate the use of this distance.

Example 4 *Let* $\boldsymbol{\sigma}^1 = (1, 2, 3, 3)$ *and* $\boldsymbol{\sigma}^2 = (2, 1, 1, 1)$ *be two non-strict rankings of four items; the bucket orders corresponding to these two rankings are* $\{\{1\}, \{2\}, \{3, 4\}\}$ *and* $\{\{2, 3, 4\}, \{1\}\}$*, respectively. The example highlights all three cases of the distance:* $K_{12}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 1$ *(Case 1),* $K_{34}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 0$ *(Case 2), and* $K_{23}^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = p$ *(Case 3). Considering all distinct item-pairs, we obtain* $K^{(p)}(\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2) = 3 + 2p$.

4.3 The Generalized Rank Aggregation Problem

The $K^{(p)}$ distance has been utilized for comparing non-strict rankings in numerous applications such as multiagent system evaluation (Rowland *et al.*, 2019), CP-nets (Loreggia *et al.*, 2018), and social network analysis (Zhang *et al.*, 2018). In its general form, the distance has not been studied in the context of ranking aggregation; however, apart from KEMENY-AGG, Brancotte *et al.* (2015) and Andrieu *et al.* (2021) have used this distance for the special case induced by fixing the penalty parameter to $p = 1$. Section 4.3.1 formally defines RANK-AGG, proves it is NP-hard and proposes an exact formulation of this problem. Section 4.3.2 devises a constraint relaxation

method for solving the formulation more efficiently. Finally, Section 4.3.3 proposes a novel heuristic algorithm.

4.3.1 Definition and Formulation

RANK-AGG seeks a ranking $\boldsymbol{\sigma}^*$—either strict or non-strict—with the lowest cumulative $K^{(p)}$ distance to all the input rankings.

Definition 14 *The optimal ranking obtained from RANK-AGG can be mathematically stated as*

$$\boldsymbol{\sigma}^* = \underset{\boldsymbol{\sigma}\in\boldsymbol{\Sigma}}{\operatorname{argmin}} \sum_{l\in\boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\sigma},\boldsymbol{\sigma}^l) = \underset{\boldsymbol{\sigma}\in\boldsymbol{\Sigma}}{\operatorname{argmin}} \sum_{l\in\boldsymbol{\mathcal{L}}} \sum_{(i,j)\in\boldsymbol{\Lambda}} K_{ij}^{(p)}(\boldsymbol{\sigma},\boldsymbol{\sigma}^l). \tag{4.2}$$

Theorem 7 *RANK-AGG is NP-hard for $m \geq 4$.*

KEMENY-AGG was shown to be NP-hard for $m \geq 4$ by an encoding of the feedback arc set problem (Bartholdi *et al.*, 1989; Dwork *et al.*, 2001). Since KEMENY-AGG is only a special case of RANK-AGG, the latter inherits the computational complexity of the former. It is pertinent to add that $m = 2$ has a trivial solution as both of the input lists are optimal solutions. The computational complexity of the feedback arc set problem and KEMENY-AGG for $m = 3$ is an open problem (Dwork *et al.*, 2001), to the best of our knowledge.

To introduce an exact formulation for RANK-AGG, the cumulative $K^{(p)}$ distance between a given ranking $\boldsymbol{\sigma} \in \boldsymbol{\sigma}$ and all the input rankings is expressed equivalently as $\sum_{(i,j)\in\boldsymbol{\Lambda}} K_{ij}^{(p)}(\boldsymbol{\sigma})$ where,

$$K_{ij}^{(p)}(\boldsymbol{\sigma}) = \begin{cases} s_{ji} + p\,t_{ij} & \text{if } i \succ_{\boldsymbol{\sigma}} j, \\ s_{ij} + p\,t_{ij} & \text{if } j \succ_{\boldsymbol{\sigma}} i, \\ p(s_{ij} + s_{ji}) & \text{if } i \approx_{\boldsymbol{\sigma}} j. \end{cases} \tag{4.3}$$

Eq. (5.7) states that, whenever item i is ranked ahead of item j in $\boldsymbol{\sigma}$, the imposed $K^{(p)}$ distance between $\boldsymbol{\sigma}$ and all the input rankings for this item-pair equals the number of input rankings where j is ranked ahead of i, plus p-times the number of input rankings where i and j are tied. Whenever the pair is tied, the imposed $K^{(p)}$ distance is p-times the number of input rankings where there is a strict ordering between i and j.

Brancotte *et al.* (2015) proposed a mixed-integer linear programming formulation for solving Problem (4.2) for $p = 1$. Herein, we revise their objective function to reflect any possible value of p as follows:

$$\min \quad \sum_{i\in\boldsymbol{\mathcal{X}}}\sum_{j\in\boldsymbol{\mathcal{X}}}\Big[(s_{ji}+p\,t_{ij})x_{i\succ j}+(s_{ij}+p\,t_{ij})x_{j\succ i}+\ p(s_{ij}+s_{ji})x_{i\approx j}\Big] \tag{4.4a}$$

$$\text{s.t.} \quad x_{i\succ j}+x_{j\succ i}+x_{i\approx j}=1 \qquad \forall i,j\in\boldsymbol{\mathcal{X}};\ \ j>i \tag{4.4b}$$

$$x_{i\succ j}-x_{k\succ j}-x_{i\succ k}\geq -1 \qquad \forall i,j,k\in\boldsymbol{\mathcal{X}};\ \ i\neq j\neq k \tag{4.4c}$$

$$2x_{i\succ j}+2x_{j\succ i}+2x_{j\succ k}+2x_{k\succ j}-x_{i\succ k}-x_{k\succ i}\geq 0 \qquad \forall i,j,k\in\boldsymbol{\mathcal{X}};\ \ k>i,j\neq k \tag{4.4d}$$

$$x_{i\succ j},\, x_{i\approx j}\in\mathbb{B} \qquad \forall i,j\in\boldsymbol{\mathcal{X}};\ \ i\neq j. \tag{4.4e}$$

Decision variable $x_{i\succ j}$ is equal to 1 if item i is ranked ahead of item j and 0 otherwise, and decision variable $x_{i\approx j}$ is equal to 1 if i and j are tied, and 0 otherwise. Objective function (4.4a) minimizes the cumulative $K^{(p)}$ distance to all the input rankings using Eq. (5.7). Constraint (4.4b) enforces that, for every distinct item-pair (i, j), either i is ranked ahead of j, j is ranked of i, or i and j are tied. Constraint (4.4c) enforces transitivity by preventing preference cycles (Fiorini and Fishburn, 2004; Grötschel *et al.*, 1984); for example, if i is ranked ahead of j, and j is ranked ahead of k, then i must be ranked of k as well. Constraint (4.4d) enforces that if i and j are tied, and j and k are tied, then i and k must be tied as well (see Yoo and Escobedo (2021) for

more types of preference cycles for non-strict rankings). Constraint (4.4e) specifies the domain of the variables.

Yoo and Escobedo (2021) reported that their formulation for KEMENY-AGG with non-strict rankings, denoted as GKBP, outperformed the variant of Formulation (4.4) induced by fixing $p = 1/2$. Because their formulation takes advantage of the specific relationship between Kemeny-Snell distance and the extended Kendall's correlation coefficient (Emond and Mason, 2002; Yoo *et al.*, 2020), GKBP cannot be directly applied to RANK-AGG. Nevertheless, inspired by its computational performance and the fact that its constraints are equivalent to the *axiomatic facet defining inequalities* of the weak order polytope (Fiorini and Fishburn, 2004; Yoo and Escobedo, 2021), we propose an alternative formulation for solving Problem (4.2) with the same set of constraints. The proposed formulation is a non-linear binary programming model and is given by:

$$\min \quad \sum_{i\in\boldsymbol{\mathcal{X}}}\sum_{j\in\boldsymbol{\mathcal{X}}}(s_{ji}+p\,t_{ij})y_{ij}+\sum_{i\in\boldsymbol{\mathcal{X}}}\sum_{j\in\boldsymbol{\mathcal{X}}:j>i}\big(p(s_{ij}+s_{ji})-s_{ij}-s_{ji}-2p\,t_{ij}\big)y_{ij}y_{ji} \tag{4.5a}$$

$$\text{s.t.} \quad y_{ij}+y_{ji}\geq 1 \qquad \forall i,j\in\boldsymbol{\mathcal{X}};\ j>i \tag{4.5b}$$

$$y_{ij}-y_{kj}-y_{ik}\geq -1 \qquad \forall i,j,k\in\boldsymbol{\mathcal{X}};\ i\neq j\neq k \tag{4.5c}$$

$$y_{ij}\in\mathbb{B} \qquad \forall i,j\in\boldsymbol{\mathcal{X}};\ i\neq j. \tag{4.5d}$$

Here, the decision variable y_{ij} is equal to 1 if item i is ranked ahead of or tied with item j, and 0 otherwise. Item i is ranked ahead of item j if $y_{ij}=1, y_{ji}=0$ (giving $y_{ij}y_{ji}=0$), and items i and j are tied whenever $y_{ij}=y_{ji}=1$ (giving $y_{ij}y_{ji}=1$). Objective function (4.5a) minimizes the cumulative $K^{(p)}$ distance to all the input rankings. Constraint (4.5b) enforces that i and j cannot be simultaneously dispreferred over each other. Constraint (4.5c) imposes transitivity, and Constraint (4.5d) specifies

the domain of variables. Let $\overline{\boldsymbol{\sigma}}$ be the corresponding non-strict ranking induced by a feasible solution to Formulation (4.5); in particular, the rank of item i in $\overline{\boldsymbol{\sigma}}$ is calculated as $\overline{\boldsymbol{\sigma}}_i := n - \sum_{j \in \boldsymbol{\mathcal{X}}: i \neq j} y_{ij}$.

The objective function (4.5a) can be linearized using a technique proposed by Glover and Woolsey (1974). For each distinct item-pair (i, j), the binary product $y_{ij}y_{ji}$ in the objective function can be replaced by the auxiliary continuous variable z_{ij}, with the addition of four constraints: $z_{ij} \leq y_{ij}$, $z_{ij} \leq y_{ji}$, $z_{ij} \geq y_{ij} + y_{ji} - 1$, $z_{ij} \geq 0$. However, since the objective coefficient of $y_{ij}y_{ji}$, i.e., $\big(p(s_{ij} + s_{ji}) - s_{ij} - s_{ji} - 2p\,t_{ij}\big)$, is always less than or equal to zero, constraint $z_{ij} \geq y_{ij} + y_{ji} - 1$ becomes redundant; that is, whenever $y_{ij} = y_{ji} = 1$, the objective function has incentive to set z_{ij} to its maximum value of 1 and there is no need for this constraint. The full mixed-integer linear program of Problem (4.2) is as follows:

$$\min \quad \sum_{i \in \boldsymbol{\mathcal{X}}} \sum_{j \in \boldsymbol{\mathcal{X}}} (s_{ji} + p\,t_{ij}) y_{ij} + \sum_{i \in \boldsymbol{\mathcal{X}}} \sum_{j \in \boldsymbol{\mathcal{X}}: j > i} \big(p(s_{ij} + s_{ji}) - s_{ij} - s_{ji} - 2p\,t_{ij}\big) z_{ij} \tag{4.6a}$$

$$\text{s.t.} \quad y_{ij} + y_{ji} \geq 1 \qquad \forall i, j \in \boldsymbol{\mathcal{X}};\ j > i \tag{4.6b}$$

$$y_{ij} - y_{kj} - y_{ik} \geq -1 \qquad \forall i, j, k \in \boldsymbol{\mathcal{X}};\ i \neq j \neq k \tag{4.6c}$$

$$z_{ij} \leq y_{ij} \qquad \forall i, j \in \boldsymbol{\mathcal{X}};\ j > i \tag{4.6d}$$

$$z_{ij} \leq y_{ji} \qquad \forall i, j \in \boldsymbol{\mathcal{X}};\ j > i \tag{4.6e}$$

$$z_{ij} \geq 0 \qquad \forall i, j \in \boldsymbol{\mathcal{X}};\ j > i \tag{4.6f}$$

$$y_{ij} \in \{0, 1\} \qquad \forall i, j \in \boldsymbol{\mathcal{X}};\ i \neq j. \tag{4.6g}$$

It is possible to derive a lower bound on Problem (4.2) using the pairwise comparison information provided in Eq. (5.7).

Proposition 5 *A lower bound on Problem* (4.2) *can be obtained as:*

$$LB = \sum_{(i,j) \in \boldsymbol{\Lambda}} \min\left(s_{ji} + p\,t_{ij},\ s_{ij} + p\,t_{ij},\ p(s_{ij} + s_{ji})\right). \tag{4.7}$$

For every distinct item-pair, Eq. (4.7) selects the smallest contribution among all three possible preference relationships between the items. Proposition 5 effectively generalizes the lower bound for KEMENY-AGG with strict rankings introduced in Davenport and Kalagnanam (2004) and with non-strict rankings introduced in Akbari and Escobedo (2021). Furthermore, this lower bound can be boosted by detecting preference cycles in the input rankings, as the solution obtained by selecting the smallest contribution for each pair of distinct items may not be transitive (Conitzer *et al.*, 2006; Milosz and Hamel, 2018). Another lower bound can be obtained by solving the linear programming relaxation of Formulations (4.4) or (4.6).

4.3.2 Constraint Relaxation Method

Formulation (4.6) has $O(n^3)$ transitivity constraints (i.e., Constraints (4.6c)) which makes solving it to optimality very difficult and practically impossible for large values of n. However, only a very small fraction of these constraints are typically necessary to solve rank aggregation models to optimality (Pedings *et al.*, 2012). Motivated by the performance of CR in accelerating lower bounding technique of KEMENY-AGG, we use CR to solve instances that are practically unsolvable with off-the-shelf methods. The pseudocode of CR is presented in Algorithm 4, which begins by dropping all transitivity constraints from Formulation (4.6)—this is denoted as the *Relaxed Formulation*. At each iteration of CR, the Relaxed Formulation is solved and the solution is inspected to determine whether there are unsatisfied transitivity constraints, which are added to the model. This process is repeated until the solution does not violate any transitivity constraints. CR is guaranteed to obtain an optimal solution, as all transitivity constraints (which are finite) are added to the Relaxed Formulation in the worst-case scenario.

Algorithm 4: Constraint Relaxation (CR) Method

Input : p, $[s_{ij}] \in \mathbb{Z}^{n.n}$, $[t_{ij}] \in \mathbb{Z}^{n.n}$

Output: Optimal solution to Formulation (4.6)

$t := 0$;

$\boldsymbol{\Xi} := \{(i,j,k) \,|\, i,j,k \in \boldsymbol{\mathcal{X}};\, i \neq j \neq k\}$; `// set of all item-triplets`

$\boldsymbol{\Xi}' := \emptyset$; `// set of item-triplets whose preference transitivity constraints are included in the Relaxed Formulation (see the next line)`

Build the Relaxed Formulation:

$$\begin{aligned} \min \quad & \sum_{i \in \boldsymbol{\mathcal{X}}} \sum_{j \in \boldsymbol{\mathcal{X}}} (s_{ji} + p\, t_{ij}) y_{ij} + \sum_{i \in \boldsymbol{\mathcal{X}}} \sum_{j \in \boldsymbol{\mathcal{X}}: j > i} \big(p(s_{ij} + s_{ji}) - s_{ij} - s_{ji} - 2p\, t_{ij}\big) z_{ij} \\ \text{s.t.} \quad & (4.6b), (4.6d) - (4.6g) \\ & y_{ij} - y_{kj} - y_{ik} \geq -1 \quad \forall (i,j,k) \in \boldsymbol{\Xi}' \end{aligned}$$

Transitivity_Violation = False;

while *Transitivity_Violation is not True* **do**

  $t \leftarrow t + 1$;

  Solve the Relaxed Formulation and obtain solution $y_{ij}^{(t)}$, where $i, j \in \boldsymbol{\mathcal{X}}$, $i \neq j$;

  for $(i,j,k) \in \boldsymbol{\Xi} \backslash \boldsymbol{\Xi}'$ **do**

    if $y_{ij}^{(t)} - y_{kj}^{(t)} - y_{ik}^{(t)} \not\geq -1$ **then**

      Transitivity_Violation = True;

      $\boldsymbol{\Xi}' \leftarrow \boldsymbol{\Xi}' \cup \{(i,j,k)\}$;

Return $\boldsymbol{\sigma}^* = [n - \sum_{j \in \boldsymbol{\mathcal{X}}: i \neq j} y_{ij}$ *for* i *in* $\boldsymbol{\mathcal{X}}]$

4.3.3 The Least Imposed Cost Heuristic (LICH)

In this section, we propose a greedy iterative algorithm, denoted as the Least Imposed Cost Heuristic (LICH), for solving RANK-AGG. Placing item i at any position of a bucket order contributes a certain amount to the objective function (4.6a); denote this imposed cost as $\upsilon(i)$. The algorithm works by iteratively adding an item among a small number of positions in a working bucket order, namely the available item with the lowest associated υ-value.

LICH's pseudocode is presented in Algorithm 7 and is summarized as follows. In

the first iteration, one item needs to be selected to initialize the working bucket order. Placing item i in the first place, assuming that it is ranked ahead of all other items, imposes the following cost:

$$\upsilon(i) = \sum_{j \in \boldsymbol{\mathcal{X}} \setminus \{i\}} s_{ji} + p\, t_{ij}.$$

A working bucket order $\boldsymbol{\mathcal{B}}$ is initialized by placing the item with the lowest imposed cost in the first bucket.

In the next iterations, the remaining items are compared with only the items in the last bucket of the working bucket order, for the sake of efficiency. At each iteration and for each remaining item i, three different imposed costs are calculated based on where i is added to the working bucket order: 1) in the last bucket, 2) a new bucket right after the last bucket, and 3) a new bucket right before the last bucket. For each item, consider the minimum of the three calculated imposed costs. Formally, let B_w be the last bucket of the working bucket order $\boldsymbol{B}$ and $\mathcal{X}^r$ be the set of remaining items to be placed in the working bucket order, calculate

$$\upsilon(i) = \min \left(\sum_{j \in B_w} p(s_{ij} + s_{ji}), \sum_{j \in B_w} (s_{ij} + p\, t_{ij}), \sum_{j \in B_w} (s_{ji} + p\, t_{ij}) \right) \quad \forall i \in \mathcal{X}^r.$$

The item with the lowest imposed cost overall is added to the working bucket order in the appropriate manner (according to the aforementioned three cases). As a post-processing subroutine, adjacent buckets are merged if doing so decreases the value of the objective function (4.6a).

Theorem 8 *Algorithm 7 has a time complexity of* $O(n^3)$.

The worst time complexity of Algorithm 7 occurs when the working bucket order has only one bucket; in this case, the last bucket of the working bucket order is always of maximum size. In this case, the number of distinct item-pairs for which we need to

Algorithm 5: The Least Imposed Cost Heuristic (LICH)

Input : $p, \boldsymbol{\mathcal{X}}, [s_{ij}] \in \mathbb{R}^{n\times n}, [t_{ij}] \in \mathbb{R}^{n\times n}$
Output: Solution non-strict ranking

$i' := \arg\min_{i\in\boldsymbol{\mathcal{X}}} \sum_{j\in\boldsymbol{\mathcal{X}}} s_{ji} + p\, t_{ij}$;
$\boldsymbol{\mathcal{B}} := \{\{i'\}\}$; `// initialize the working bucket order`
$\mathcal{X}^r := \boldsymbol{\mathcal{X}}\backslash\{i'\}$; `// set of remaining items`
for $t = 1, \ldots, n-1$ **do**
  Let B_w be the last bucket of the working bucket order $\boldsymbol{\mathcal{B}}$;
  for $i \in \mathcal{X}^r$ **do**
    $\upsilon(i) = \min\left(\sum_{j\in B_w}(s_{ji} + p\, t_{ij}), \sum_{j\in B_w}(s_{ij} + p\, t_{ij}), \sum_{j\in B_w} p(s_{ij} + s_{ji})\right)$;
    $i' = \arg\min_{i\in\mathcal{X}^r} \upsilon(i)$; `// find the item with the lowest imposed cost`
    $\mathcal{X}^r \leftarrow \mathcal{X}^r\backslash\{i'\}$; `// remove` i' `from the set of remaining items`
    `// The next block of code adds` i' `to the working bucket order in a way that it induces the lowest imposed cost`
    if $\upsilon(i') = \sum_{j\in B_w} p(s_{i'j} + s_{ji'})$ **then**
      $\boldsymbol{\mathcal{B}} \leftarrow \{B_1, \ldots, B_{w-1}, B_w \cup \{i'\}\}$;
    if $\upsilon(i') = \sum_{j\in B_w} (s_{i'j} + pt_{i'j})$ **then**
      $\boldsymbol{\mathcal{B}} \leftarrow \{B_1, \ldots, B_{w-1}, B_w, \{i'\}\}$;
    if $\upsilon(i') = \sum_{j\in B_w} (s_{ji'} + pt_{i'j})$ **then**
      $\boldsymbol{\mathcal{B}} \leftarrow \{B_1, \ldots, B_{w-1}, \{i'\}, B_w\}$
Merge adjacent buckets of $\boldsymbol{\mathcal{B}}$ if doing so improves the value of objective function (4.6a);
Obtain $\boldsymbol{\sigma}$ from $\boldsymbol{\mathcal{B}}$ (as explained in Section 4.2);
Return $\boldsymbol{\sigma}$;

calculate the imposed cost is given by

$$n(n-1) + (n-1)(1) + (n-2)(2) + \cdots + (1)(n-1) = n(n-1) + \sum_{i=1}^{n-1}(n-i)i$$
$$= n(n-1) + \frac{1}{6}(n-1)n(n+1)$$

The imposed costs of each item-pair can be obtained in constant time. Therefore,

the complexity of the full algorithm is $O(n^3)$. Note that the worst time complexity of Algorithm 7 occurs when at least $n-1$ items are tied in the optimal ranking, and its time complexity reduces to $O(n^2)$ in the case of strict rankings, as all buckets are singletons in this case.

4.4 Generalizing the Condorcet Criterion and its Variants

XCC and NXCC have been defined only for KEMENY-AGG. This subsection expands the concept of Condorcet partitions to RANK-AGG. To that end, it first redefines the concept of pairwise preference to adapt to the nature of the generalized problem, and it introduces a novel social choice property termed as the Generalized Extended Condorcet Criterion (GXCC).

Definition 15 *Item i is (strictly) pairwise preferred over item j if*

$$s_{ij} > \max\Big(\Big(\frac{1-p}{p}\Big)s_{ji} + t_{ij},\, s_{ji}\Big),$$

and it is weakly pairwise preferred over j if

$$s_{ij} \geq \max\Big(\Big(\frac{1-p}{p}\Big)s_{ji} + t_{ij},\, s_{ji}\Big).$$

Definition 16 *(GXCC) Given a fixed penalty parameter p, assume that $\boldsymbol{\mathcal{X}}$ can be arranged into a partition $\boldsymbol{X}^{(p)} = \{X_1, X_2, \ldots, X_w\}$ such that*

$$s_{ij} > \max\Big(\Big(\frac{1-p}{p}\Big)s_{ji} + t_{ij}, s_{ji}\Big) \quad \forall i \in X_k,\ \forall j \in X_{k'},\ \forall k < k'. \tag{4.8}$$

GXCC specifies that $\boldsymbol{\sigma}^*$ must rank all items in lower-indexed subsets of $\boldsymbol{X}^{(p)}$ ahead of all items in higher-indexed subsets. That is, when (4.8) *holds, then*

$$i \succ_{\boldsymbol{\sigma}^*} j \quad \forall i \in X_k,\ \forall j \in X_{k'},\ \forall k < k'.$$

GXCC contains XCC and NXCC as special cases: it becomes NXCC when $p = 1/2$, and it becomes XCC when the same penalty is used and all the input rankings are

strict.

To explain the rationale for Definition 16, recall that in KEMENY-AGG item i is said to be pairwise preferred by a decisive majority over item j if the number of input rankings who rank i ahead of j is greater than those who rank j ahead of i, plus who those tie them. This is related to the Kemeny-Snell distance's fixed penalty of 1/2 for every partial rank reversal between the solution ranking and the input rankings and NXCC's consideration of both full and partial rank reversals to determine when j cannot be ahead of i. However, in RANK-AGG every partial rank reversal yields penalty p. The coefficient $\frac{1-p}{p}$ accounts for the contribution of every full and every partial rank reversal in the overall decisive majority associated with the generalized Kendall tau distance.

The following theorem proves that the optimal solutions to RANK-AGG are consistent with GXCC. This means that solving the subproblems induced by the subsets of the GXCC partition independently to optimality and then concatenating the results in the proper order (placing all items in the lower-indexed subsets ahead of all items in the higher-indexed subsets) is guaranteed to yield an optimal solution to RANK-AGG. To the best of our knowledge, this is the first time exact Condorcet partitioning scheme has been applied to a problem other than KEMENY-AGG in its general form.

Theorem 9 *RANK-AGG satisfies GXCC.*

We use contradiction. Without loss of generality, let $\boldsymbol{X}^{(p)} = \{\overline{X}, \overline{X}^c\}$ be a GXCC bipartition of $\boldsymbol{\mathcal{X}}$, where $\overline{X}^c = \boldsymbol{\mathcal{X}} \backslash \overline{X}$, and let $\boldsymbol{\pi}^*$ be an optimal ranking where at least one item in $\overline{X}^c$ is ranked ahead of or tied with at least one item in $\overline{X}$. Consider a ranking $\boldsymbol{\pi}'$ obtained by modifying $\boldsymbol{\pi}^*$ such that all items of $\overline{X}$ are ranked ahead of all items in $\overline{X}^c$, and the relative orderings of all items within $\overline{X}$ and $\overline{X}^c$ are as in $\boldsymbol{\pi}^*$.

The difference between the cumulative $K^{(p)}$ distances (i.e., to all the input rankings) accrued with $\boldsymbol{\pi}^*$ versus $\boldsymbol{\pi}'$, denoted by Δ, is given by

$$\begin{aligned}\Delta &= \sum_{l\in\boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) - \sum_{l\in\boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\pi}', \boldsymbol{\pi}^l) \\ &= \sum_{i\in\overline{X}} \sum_{j\in\overline{X}^c} \sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) - \sum_{i\in\overline{X}} \sum_{j\in\overline{X}^c} \sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}', \boldsymbol{\pi}^l) \\ &= \sum_{i\in\overline{X}} \sum_{j\in\overline{X}^c} \sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) - \sum_{i\in\overline{X}} \sum_{j\in\overline{X}^c} (s_{ji} + p\, t_{ij}).\end{aligned}$$

The last equation comes from the starting assumption that $\boldsymbol{\pi}'$ ranks all items in $\overline{X}$ ahead of all items in $\overline{X}^c$. Therefore, the contribution of every distinct item-pair (i, j) where $i \in \overline{X}$ and $j \in \overline{X}^c$ in $\sum_{l\in\boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\pi}', \boldsymbol{\pi}^l)$ is equal to $s_{ji} + p\, t_{ij}$. Observe that item-pairs from different subsets do not contribute to Δ as their relative orderings are the same in $\boldsymbol{\pi}^*$ and $\boldsymbol{\pi}'$. To determine the sign of Δ, we compare the terms $\sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l)$ and $s_{ji} + p\, t_{ij}$. From Eq. (5.7), for every distinct item-pair (i, j), $i \in \overline{X}$ and $j \in \overline{X}^c$, we have

$$\sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) = \begin{cases} s_{ji} + p\, t_{ij} & \text{if } i \succ_{\boldsymbol{\pi}^*} j, \\ s_{ij} + p\, t_{ij} & \text{if } j \succ_{\boldsymbol{\pi}^*} i, \\ p(s_{ij} + s_{ji}) & \text{if } i \approx_{\boldsymbol{\pi}^*} j. \end{cases} \tag{4.9}$$

Next, we show that for such a distinct item-pair, $s_{ji} + p\, t_{ij}$ does not exceed $\sum_{l\in\boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l)$ in cases where $j \succ_{\boldsymbol{\pi}^*} i$ and $i \approx_{\boldsymbol{\pi}^*} j$ ($K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l)$ equals $s_{ji} + p\, t_{ij}$ when $i \succ_{\boldsymbol{\pi}^*} j$). Based on the starting assumption that $\boldsymbol{X}^{(p)}$ satisfies GXCC, for every distinct item-pair (i, j), $i \in \overline{X}$ and $j \in \overline{X}^c$, the following inequalities can be derived

$$s_{ij} > \max\Big(\Big(\frac{1-p}{p}\Big)s_{ji} + t_{ij},\, s_{ji}\Big) \Longrightarrow s_{ij} > \Big(\frac{1-p}{p}\Big)s_{ji} + t_{ij} \Longrightarrow p(s_{ij} + s_{ji}) > s_{ji} + p\, t_{ij}.$$

Furthermore, we have

$$s_{ij} > \max\left(\left(\frac{1-p}{p}\right)s_{ji} + t_{ij},\, s_{ji}\right) \implies s_{ij} > s_{ji} \implies s_{ij} + p\,t_{ij} > s_{ji} + p\,t_{ij}.$$

Therefore, for every distinct item-pair (i, j), where $i \in \overline{X}$ and $j \in \overline{X}^c$, we have

$$\sum_{l \in \boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) \geq s_{ji} + p\,t_{ij}, \tag{4.10}$$

and summing over all distinct item-pairs (i, j), where $i \in \overline{X}$ and $j \in \overline{X}^c$, gives

$$\sum_{i \in \overline{X}} \sum_{j \in \overline{X}^c} \sum_{l \in \boldsymbol{\mathcal{L}}} K_{ij}^{(p)}(\boldsymbol{\pi}^*, \boldsymbol{\pi}^l) \geq \sum_{i \in \overline{X}} \sum_{j \in \overline{X}^c} s_{ji} + p\,t_{ij}.$$

The above inequality implies that $\Delta \geq 0$. According to the given assumption, there exists at least one item in $\overline{X}^c$ that is ranked ahead of or tied with at least one item in $\overline{X}$. Hence , Eq. (4.10) holds strictly for at least one item-pair, meaning that $\Delta > 0$, which contradicts the optimality of $\boldsymbol{\pi}^*$. Therefore, we can conclude that all items in $\overline{X}$ must be ranked ahead of all items in $\overline{X}^c$ in the optimal ranking.

Finally, we extend the proof to the case with $|\boldsymbol{X}^{(p)}| = w > 2$. Consider a GXCC bipartition $\boldsymbol{X}^{(p)} = \{\overline{X}, \overline{X}^c\}$ where $\overline{X} = X_1$ and $\overline{X}^c = \{X_2, X_3, \ldots, X_w\}$. Applying the prior result, all items in X_1 must be ranked ahead of all items in $\boldsymbol{\mathcal{X}} \backslash X_1$. Next, consider bipartition $\boldsymbol{X}^{(p)} = \{\overline{X}, \overline{X}^c\}$ where $\overline{X} = \{X_1, X_2\}$ and $\overline{X}^c = \{X_3, \ldots, X_w\}$; from the preceding case, all items in X_1 and X_2 must be ranked ahead of all items in $\boldsymbol{\mathcal{X}} \backslash (X_1 \cup X_2)$, and all items in X_1 must be ranked ahead of all items in X_2. Continuing in this manner, the only way that this statement holds for all bipartitions of the form $\boldsymbol{X}^{(p)} = \{\{X_1, \ldots, X_k\}, \{X_{k+1}, \ldots, X_w\}\}$, where $k \in \{1, \ldots, w-1\}$, is if $\boldsymbol{\pi}^*$ ranks all items in lower-indexed subsets of $\boldsymbol{X}^{(p)}$ ahead of all items in higher-indexed subsets.

4.4.1 Enlarged GXCC Partitions

Let $\wp(\boldsymbol{\mathcal{X}})$ denote the class of partitions that satisfy GXCC and consider the case when there are multiple optimal rankings. The fact that all optimal rankings must be consistent with any $\boldsymbol{X} \in \wp(\boldsymbol{\mathcal{X}})$ can be viewed as a restrictive condition. It might be possible to make the partition finer, i.e., one with more subsets, by requiring that it respects at least one rather than all of the optimal rankings. We refer to such partitions as *enlarged GXCC partitions*, denoted as $\boldsymbol{X}_e$. Schalekamp and Zuylen (2009) defined a type of enlarged XCC partitions for strict rankings as follows. Assume that $\boldsymbol{\mathcal{X}}$ can be arranged into a partition $\boldsymbol{X}_e = \{X_1, X_2, \ldots, X_w\}$ such that $s_{ij} \geq s_{ji} \, \forall i \in X_k \forall j \in X_{k'}, \forall k < k'$. Then, solving the subsets of $\boldsymbol{X}_e$ independently and concatenating the results will respect at least one of the optimal strict rankings. It is possible to extend this idea to GXCC to obtain possibly more effective partitions.

Corollary 1 *(Enlarged GXCC) Given a fixed penalty parameter p, assume that $\boldsymbol{\mathcal{X}}$ can be arranged into a partition $\boldsymbol{X}_e^{(p)} = \{X_1, X_2, \ldots, X_w\}$ such that*

$$s_{ij} \geq \max\left(\left(\frac{1-p}{p}\right)s_{ji} + t_{ij}, \, s_{ji}\right) \;\; \forall i \in X_k, \;\; \forall j \in X_{k'}, \;\; \forall k < k'. \tag{4.11}$$

Then, there exists at least one optimal ranking in which all items in the lower-indexed subsets of $\boldsymbol{X}_e^{(p)}$ are ranked ahead of all items in its higher-indexed subsets. That is, when (4.11) *holds,*

$$\exists \boldsymbol{\sigma}^* \in \boldsymbol{\Pi} : i \succ_{\boldsymbol{\sigma}^*} j \;\; \forall i \in X_k, \;\; \forall j \in X_{k'}, \;\; \forall k < k'.$$

The rationale mirrors that of the proof of Theorem 9 up to the point where it concludes that $\Delta \geq 0$. Applying those steps, since $\boldsymbol{\sigma}^*$ is an optimal ranking according to the starting assumption and $\Delta \geq 0$, it can be concluded that $\boldsymbol{\sigma}'$ is an optimal ranking as well. Notice that enlarged GXCC requires items in the lower-indexed subsets to be only weakly pairwise preferred over items in the higher-indexed subsets.

Example 5 *Consider an instance with 10 rankings of 6 items. The input rankings and the pairwise comparison matrices,* $\boldsymbol{S} = [s_{ij}] \in \mathbb{Z}^{6\times 6}$ *and* $\boldsymbol{T} = [t_{ij}] \in \mathbb{Z}^{6\times 6}$*, are given by*

Item	Input Rankings									
	$\boldsymbol{\sigma}^1$	$\boldsymbol{\sigma}^2$	$\boldsymbol{\sigma}^3$	$\boldsymbol{\sigma}^4$	$\boldsymbol{\sigma}^5$	$\boldsymbol{\sigma}^6$	$\boldsymbol{\sigma}^7$	$\boldsymbol{\sigma}^8$	$\boldsymbol{\sigma}^9$	$\boldsymbol{\sigma}^{10}$
1	2	4	2	2	3	4	5	2	1	5
2	3	1	3	5	1	3	5	1	1	5
3	3	3	1	3	1	1	2	5	2	3
4	1	3	1	1	5	1	1	5	3	1
5	4	5	4	1	2	2	3	4	4	2
6	5	2	5	4	4	5	4	3	5	4

$$\mathbf{S} = \begin{bmatrix} 0 & 3 & 4 & 3 & 5 & 7 \\ 4 & 0 & 3 & 4 & 6 & 7 \\ 6 & 5 & 0 & 2 & 7 & 8 \\ 7 & 6 & 4 & 0 & 7 & 7 \\ 5 & 4 & 3 & 2 & 0 & 8 \\ 3 & 3 & 2 & 3 & 2 & 0 \end{bmatrix}, \mathbf{T} = \begin{bmatrix} 0 & 3 & 0 & 0 & 0 & 0 \\ 3 & 0 & 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The standard GXCC partitions for $p = 1/2, 3/4,$ *and* 1 *are given by* $\boldsymbol{X}^{(1/2)} = \{\{1,2,3,4,5\},\{6\}\}$*,* $\boldsymbol{X}^{(3/4)} = \{\{3,4\},\{1,2,5\},\{6\}\}$*, and* $\boldsymbol{X}^{(1)} = \{\{3,4\},\{2\},\{1,5\},\{6\}\}$*, respectively. The enlarged GXCC partitions are given by* $\boldsymbol{X}_e^{(1/2)} = \{\{3,4\},\{1,2\},\{5\},\{6\}\}$*,* $\boldsymbol{X}_e^{(3/4)} = \{\{3,4\},\{2\},\{5\},\{1\},\{6\}\}$*, and* $\boldsymbol{X}_e^{(1)} = \{\{4\},\{3\},\{2\},\{5\},\{1\},\{6\}\}$*.*

Example 5 illustrates the improved practicality of enlarged GXCC partitions. Considering the enlarged GXCC partitions for $p = 1/2$, only the relative ordering of item-pairs $(3,4)$ and $(1,2)$ needs to be determined; for $p = 3/4$, only the relative ordering of item-pair $(3,4)$ needs to be determined; and for $p = 1$, an optimal solution is trivially obtained from the partition. Clearly this accelerates the solution to RANK-AGG as each of enlarged GXCC partitions improves on its standard counterpart.

Due to the enhanced practicality of enlarged GXCC, we focus on this partitioning mechanism for the rest of the paper. To obtain an enlarged GXCC partition, we modify the algorithm of Yoo and Escobedo (2021), which conducts NXCC partitioning by performing sequential pairwise comparisons. The modified algorithm is presented in Algorithm 6. It starts by placing the first item in a subset of the working partition. Then, it adds exactly one item to the working partition at each iteration. Let item i

Algorithm 6: Enlarged GXCC Partitioning

Input : p, $[s_{ij}] \in \mathbb{Z}^{n\times n}$, $[t_{ij}] \in \mathbb{Z}^{n\times n}$
Output : Enlarged GXCC Partition

$\boldsymbol{X}_e^{(p)} = \{\{1\}\}$;
for $i = 2$ **to** $|\boldsymbol{\mathcal{X}}|$ **do**
  $k = 1$;
  if $\forall j \in X_1$ *s.t.,* $s_{ij} \geq \max\left(\left(\frac{1-p}{p}\right) s_{ji} + t_{ij}, s_{ji}\right)$ **then**
    Insert i before X_1, and increment the index of subsets after $X_{\kappa(i)}$ by 1;
    $k \leftarrow 3$;
  else if $\forall j \in X_1$ *s.t.,* $s_{ji} \geq \max\left(\left(\frac{1-p}{p}\right) s_{ij} + t_{ij}, s_{ji}\right)$ **then**
    Insert i after X_1, and increment the index of subsets after $X_{\kappa(i)}$ by 1;
    $k \leftarrow 3$;
  else
    Insert i in X_1;
    $k \leftarrow 2$;
  while $k \leq |\boldsymbol{\mathcal{X}}|$ **do**
    if $\forall j \in X_k$ *s.t.,* $s_{ij} \geq \max\left(\left(\frac{1-p}{p}\right) s_{ji} + t_{ij}, s_{ji}\right)$ **then**
      $k \leftarrow k + 1$;
    else if $\forall j \in X_k$ *s.t.,* $s_{ji} \geq \max\left(\left(\frac{1-p}{p}\right) s_{ij} + t_{ij}, s_{ij}\right)$ **then**
      if $|\kappa(i) - k| = 1$ *and* $|X_{\kappa(i)}| = 1$ **then**
        Move $X_{\kappa(i)}$ after X_k;
      else
        Merge subsets from $X_{\kappa(i)}$ to X_k;
        Decrease the index of subsets after X_k by $(k - \kappa(i))$;
        $k \leftarrow \kappa(i) + 1$;
    else if $\exists j \in X_k$ *s.t.,* $s_{ji} > \max\left(\left(\frac{1-p}{p}\right) s_{ij'} + t_{ij'}, s_{ij}\right)$ **then**
      Merge subsets from $X_{\kappa(i)}$ to X_k;
      Decrease the index of subsets after X_k by $(k - \kappa(i))$;
      $k \leftarrow \kappa(i) + 1$;
Return $\boldsymbol{X}_e^{(p)}$;
* $\kappa(i)$ is the index of the subset containing item i.

denote the added item at any iteration and $\boldsymbol{X}_e^{(p)} = \{X_1, \ldots, X_w\}$ denote the working partition. The algorithm compares i with all items in the first subset of the working partition, i.e., X_1, leading to three possible outcomes. If item i is weakly pairwise preferred over all items in X_1, it is placed in a new subset right before X_1; if all items in X_1 are weakly pairwise preferred over i, i is placed in a new subset right after X_1; otherwise, it is placed in X_1. Subsequently, the algorithm iteratively checks

whether the current working partition is a valid enlarged GXCC partition by validating whether all items in lower-indexed subsets are weakly pairwise preferred over all items in higher-indexed subsets. Whenever violations are detected, the respective subsets are merged/moved until there are no violations. The algorithm has a time complexity of $O(n^2)$ (Yoo and Escobedo, 2021)

Note that GXCC and enlarged GXCC rely only on parameters used by the exact formulations; this fact, coupled with the quadratic time complexity of its algorithm, makes enlarged GXCC a great and fast pre-processing step for solving RANK-AGG via exact and heuristic methods.

We close this section by comparing GXCC and enlarged GXCC with the exact graph-based partitioning scheme of Andrieu *et al.* (2021) for the special case of RANK-AGG induced by setting $p = 1$. Upon close inspection, their method's required conditions translate to a relaxed version of GXCC but stricter than enlarged GXCC. To elaborate, this method requires all items in lower-indexed subsets to be weakly pairwise preferred (according to Definition 15 induced by setting $p = 1$) over all items in the higher-indexed subsets except for adjacent subsets, for which a strict pairwise preference is required. This relaxed version of GXCC will respect all the optimal solutions (Andrieu *et al.*, 2021); however, it is more restrictive that enlarged GXCC, since the latter respects at least one but not necessarily all optimal solutions. Nonetheless, enlarged GXCC is guaranteed to have at least as many subsets as Andrieu *et al.* (2021)'s method.

4.5 Effect of Modifying the Penalty Parameter

The value of penalty parameter p can have a big impact on the outcome of RANK-AGG, as it can alter the optimal ranking and the very nature of the problem. Consider two extreme values of p over which $K^{(p)}$ is a distance metric, namely 1/2 and 1. For

$p = 1/2$, this induces KEMENY-AGG, in which every full rank reversal of item-pair (corresponding to case 1 of the $K^{(p)}$ distance) has twice the weight of every partial rank reversal (corresponding to case 3). As the value of p increases, the weight of a partial rank reversal increases; finally, for $p = 1$, a partial rank reversal has the same weight as a full rank reversal.

Beyond the fact that $p = 1/2$ is the most frequently used value, there have been no attempts to guide the choice of p or to analyze its implications. The ensuing paragraphs provide useful insights regarding the impact of p of the resulting GXCC and enlarged GXCC partitions and on the optimal ranking.

Proposition 6 *The set of optimal objective values of RANK-AGG for all values of p forms a piecewise linear envelope.*

The cumulative $K^{(p)}$ distance between any solution ranking $\boldsymbol{\sigma} \in \boldsymbol{\sigma}$ and all the input rankings, i.e., $\sum_{l \in \boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\sigma}, \boldsymbol{\sigma}^l)$, can be expressed as $a^{(\boldsymbol{\sigma})} + pb^{(\boldsymbol{\sigma})}$, which is an affine function in terms of p, where

$$a^{(\boldsymbol{\sigma})} = \sum_{(i,j) \in \boldsymbol{\Lambda}} \Big(s_{ij} \mathbb{1}_{j \succ_{\boldsymbol{\sigma}} i} + s_{ji} \mathbb{1}_{i \succ_{\boldsymbol{\sigma}} j} \Big)$$

and

$$b^{(\boldsymbol{\sigma})} = \sum_{(i,j) \in \boldsymbol{\Lambda}} \Big((s_{ij} + s_{ji}) \mathbb{1}_{i \approx_{\boldsymbol{\sigma}} j} + t_{ij} (\mathbb{1}_{j \succ_{\boldsymbol{\sigma}} i} + \mathbb{1}_{i \succ_{\boldsymbol{\sigma}} j}) \Big).$$

Here, the function $\mathbb{1}_v$ returns 1 if v is true, and 0 otherwise; $a^{(\boldsymbol{\sigma})}$ is the number of full rank reversals between $\boldsymbol{\sigma}$ and all the input rankings, and $b^{(\boldsymbol{\sigma})}$ is the number of partial rank reversals. More specifically, if items i and j are tied in $\boldsymbol{\sigma}$, $b^{(\boldsymbol{\sigma})}$ counts the number of input rankings in which i and j are not tied; conversely, if there is a strict ordering between i and j in $\boldsymbol{\sigma}$, $b^{(\boldsymbol{\sigma})}$ counts the number of input rankings where they are tied. Since the objective function values can be expressed as a series of affine functions, and the $K^{(p)}$ distance is non-decreasing in p (Fagin *et al.*, 2004), the set of

optimal objective values for all values of p forms a piecewise linear envelope. Fig. 4.1 illustrates an example of RANK-AGG with two items. There are three possible rankings $\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2, \boldsymbol{\sigma}^3$, whose respective affine functions are displayed; the piecewise linear envelope is shown in red. Proposition 6 will be used to derive additional insights regarding the effect of penalty parameter p.

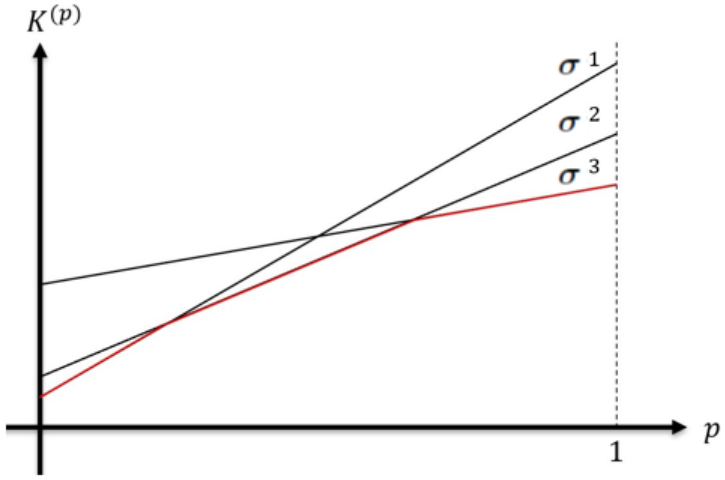

Figure 4.1: Example Depiction of Objective Function Values Obtained over All Values of p by Three Different Solution Non-strict Rankings (the Piecewise Linear Envelope Is Shown in Bolded Red)

Corollary 2 *If $\boldsymbol{\sigma}^*$ is the optimal ranking for two distinct penalty parameters p^1 and p^2 such that $0 \leq p^1 < p^2 \leq 1$, then $\boldsymbol{\sigma}^*$ is also the optimal rankings for any $p^1 < p < p^2$. Furthermore, if $\boldsymbol{\sigma}^*$ is the optimal ranking for p^1 but not for p^2, it will not be the optimal ranking for any penalty parameter $p > p^2$.*

Corollary 2 is a direct outcome of Proposition 6. This corollary can help overcome the difficulty of selecting the exact value of penalty p in certain instances. For example, if $\boldsymbol{\sigma}^*$ is the optimal ranking for $p = 1/2$ and $p = 1$, then it is also the optimal ranking for every intermediate value.

Additionally, we show that using $p = 3/4$ has an interesting interpretation, as it produces a robust solution. In particular, one may also be interested in finding the

optimal ranking with the minimum *average* $K^{(p)}$ distance to the input rankings over all possible values of p for which the resulting function is a metric, i.e., $\forall p \in [1/2, 1]$, instead of only one specific value.

Proposition 7 *The optimal ranking obtained by using $p = 3/4$ has the least average cumulative $K^{(p)}$ distance to the input rankings over the interval of penalty parameter p, where $K^{(p)}$ is a distance metric.*

Since all values of p are given the same weight, p can be treated as a random variable with a continuous uniform distribution over $[\frac{1}{2}, 1]$. Hence, the problem of finding a ranking with the least average cumulative $K^{(p)}$ distance to all the input rankings with respect to all values of $p \in [\frac{1}{2}, 1]$ is equivalent to

$$\begin{aligned}
\boldsymbol{\sigma}^* &= \arg\min_{\boldsymbol{\sigma} \in \boldsymbol{\sigma}} \ \mathbb{E}_{p \in [\frac{1}{2}, 1]} \left[\sum_{l \in \boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\sigma}, \boldsymbol{\sigma}^{\boldsymbol{l}}) \right] \\
&= \arg\min_{\boldsymbol{\sigma} \in \boldsymbol{\sigma}} \ \mathbb{E}_{p \in [\frac{1}{2}, 1]} \left[a^{(\boldsymbol{\sigma})} + p b^{(\boldsymbol{\sigma})} \right] \\
&= \arg\min_{\boldsymbol{\sigma} \in \boldsymbol{\sigma}} \ \left[a^{(\boldsymbol{\sigma})} + b^{(\boldsymbol{\sigma})} \mathbb{E}_{p \in [\frac{1}{2}, 1]}(p) \right] \\
&= \arg\min_{\boldsymbol{\sigma} \in \boldsymbol{\sigma}} \ \left[a^{(\boldsymbol{\sigma})} + 3/4 b^{(\boldsymbol{\sigma})} \right] \\
&= \arg\min_{\boldsymbol{\sigma} \in \boldsymbol{\sigma}} \ \sum_{l \in \boldsymbol{\mathcal{L}}} K^{(3/4)}(\boldsymbol{\sigma}, \boldsymbol{\sigma}^{l}).
\end{aligned}$$

As the last insight, when p increases, the cardinality of the GXCC and enlarged GXCC partitions may at times increase, but it cannot decrease.

Proposition 8 *Consider two fixed penalty parameters p_1, p_2, with $0 < p_1 < p_2 \leq 1$. For penalty parameter p_2, the GXCC and enlarged GXCC partitions have at least as many subsets as their respective partitions with penalty parameter p_1. That is $|\boldsymbol{X}^{(p_2)}| \geq |\boldsymbol{X}^{(p_1)}|$ and $|\boldsymbol{X}_e^{(p_2)}| \geq |\boldsymbol{X}_e^{(p_1)}|$.*

For every item pair $(i, j) \in \boldsymbol{\Lambda}$, we have

$$s_{ij} > \max\Big(\Big(\frac{1 - p_1}{p_1}\Big) s_{ji} + t_{ij},\, s_{ji}\Big) \geq \max\Big(\Big(\frac{1 - p_2}{p_2}\Big) s_{ji} + t_{ij},\, s_{ji}\Big).$$

Therefore, if i is pairwise preferred over j using penalty parameter p_1, it will also be pairwise preferred over j using the penalty parameter p_2. Hence, $\boldsymbol{X}^{(p_1)}$ is also a valid GXCC partition for RANK-AGG using penalty parameter p_2. As a result, $\boldsymbol{X}^{(p_2)}$ will have at least as many subsets as $\boldsymbol{X}^{(p_1)}$. A parallel set of arguments can be applied to enlarged GXCC partitions. Proposition 8 indicates that partitioning may have a more impact on large values of p. The possible effect of increasing p on the cardinality of the GXCC and enlarged GXCC partitions is demonstrated in Example 5, where $|\boldsymbol{X}^{(1/2)}| = 2$, $|\boldsymbol{X}^{(3/4)}| = 3$, $|\boldsymbol{X}^{(1)}| = 4$, and $|\boldsymbol{X}_e^{(1/2)}| = 4$, $|\boldsymbol{X}_e^{(3/4)}| = 5$, $|\boldsymbol{X}_e^{(1)}| = 6$.

4.6 Computational Results

This section performs computational studies to: 1) compare the solution times of the revised Brancotte *et al.* (2015) formulation (Formulation (4.4)), the proposed formulation (Formulation (4.6)), and the CR method; 2) investigate the effect of enlarged GXCC partitioning on the solution times of the proposed formulation and the CR method; 3) evaluate the performance of the proposed heuristic, both in terms of solution quality and run time; and 4) investigate the effect of enlarged GXCC partitioning on the solution quality and run time of the heuristic method.

For all tested instances, we use three penalty values $p \in \{1/2, 3/4, 1\}$. All experiments herein were carried out on a PC with an Intel(R) Xeon(R) CPU E5-2680 2.40 GHz with 64 GB RAM. All optimization models were solved using CPLEX solver version 20.1, with a time limit of 7,200 seconds. The %Deviation from optimality of LICH is calculated as

$$\%Deviation = \frac{\text{objective function value of LICH - optimal objective function value}}{\text{optimal objective function value}}.$$

For the remainder of this section and the associated tables, the revised Brancotte *et al.* (2015) binary programming formulation is denoted as BBP, and the proposed

mixed-integer programming formulation is denoted as MIP. Additionally, the solution method consists of solving the partitioned problem via enlarged GXCC and then MIP as GXCC_MIP, solving the partitioned problem via enlarged GXCC and then CR as GXCC_CR, and solving the partitioned problem via enlarged GXCC and then LICH as GXCC_LICH.

The experiments consider two real-world data sets introduced in Section 3.2.2: the Biomedical data set (Cohen-Boulakia *et al.*, 2011) and the TOC - "Orders with Ties - Complete List" data set from Preflib (Mattei and Walsh, 2013). From TOC, only those instances with 40 to 351 items are used, as other instances of this data set are either too small and easy to solve or too large to be solved using exact methods. First, we compare the solution times of the exact methods, beginning with the results of the TOC data set reported in Table 4.1; the best solution time(s) attained for each instance and each of tested values of p is shown in bold. On average, MIP and CR were more than 2x and 12x faster than BBP, respectively. In fact, BBP had a higher run time than MIP, and MIP than CR, for each of tested instances and values of p. BBP failed to obtain the optimal solution of "ED-10-50" for $p = 1/2$ and $p = 1$ within the two-hour time limit; however, MIP and CR were able to solve these two cases in less than four minutes. Additionally, "ED-10-49" could not be directly solved via BBP and MIP due to out-of-memory errors, however, CR was able to solve it to optimality. Table 4.2 reports the solution times of the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set, where a similar pattern can be observed; the best solution time(s) attained for each instance and each of tested values of p is shown in bold. BBP had a higher run time than MIP, and MIP had a higher run time than CR for each of tested instances and each of tested values of p, except in one case. Additionally, BBP failed to obtain the optimal solution of "Prostate Cancer" for $p = 1/2$ and $p = 1$ within the two-hour time limit; however,

Table 4.1: Solution Time (in Seconds) of Different Exact Methods with and Without Prior GXCC Partitioning for Different Values of p for TOC Instances With $40 \leq n \leq 400$

Instance id[%]	n	$p = 1/2$					$p = 3/4$					$p = 1$				
		BBP	MIP	CR	GXCC_MIP	GXCC_CR	BBP	MIP	CR	GXCC_MIP	GXCC_CR	BBP	MIP	CR	GXCC_MIP	GXCC_CR
ED-10-21	40	3.34	1.76	1.04	0.47	**0.97**	3.41	1.85	1.08	**0.13**	0.51	3.38	1.81	0.63	**0.12**	0.21
ED-10-22	40	3.79	1.74	**1.37**	1.74	**1.37**	3.41	1.75	**1.13**	1.75	**1.13**	3.36	1.73	**0.99**	1.73	**0.99**
ED-10-30	40	3.56	1.86	0.90	**0.09**	0.34	3.38	1.69	1.00	**0.09**	0.40	3.28	1.74	0.55	0.20	**0.13**
ED-10-20	41	3.87	1.93	1.41	**0.34**	0.80	3.67	1.90	1.70	**0.23**	0.90	3.70	1.94	0.83	**0.34**	0.68
ED-10-31	41	3.88	1.98	1.37	**0.18**	0.90	3.64	1.89	1.39	**0.31**	0.85	3.53	1.90	1.03	**0.18**	0.29
ED-10-4	42	4.25	2.05	1.28	1.40	**1.18**	3.83	2.05	1.24	1.42	**1.07**	3.87	2.05	1.01	**0.47**	0.76
ED-10-09	42	3.94	2.01	**0.69**	1.01	0.81	3.94	2.06	**0.73**	0.87	0.98	3.94	2.00	0.96	**0.84**	0.85
ED-10-06	43	4.65	2.17	**0.94**	2.06	1.19	4.18	2.20	1.17	2.06	**1.15**	4.18	2.29	1.35	2.06	**1.29**
ED-10-10	43	4.81	2.15	**1.02**	2.15	**1.02**	4.25	2.13	**0.97**	2.13	**0.97**	4.29	2.18	1.12	**0.55**	0.70
ED-10-08	44	4.56	2.34	1.71	**0.70**	1.13	4.53	2.32	1.28	**0.74**	1.47	4.44	2.30	1.55	0.76	**0.70**
ED-10-12	44	4.94	2.42	**0.98**	1.86	1.43	4.40	2.45	1.16	**0.36**	1.20	4.52	2.49	1.61	**0.50**	0.96
ED-10-13	44	4.42	2.5	**0.71**	0.95	0.81	4.44	2.30	**0.61**	1.02	0.78	4.42	2.39	0.99	0.97	**0.81**
ED-10-34	46	5.88	2.57	1.46	**0.68**	1.15	5.20	2.61	0.97	**0.70**	1.56	5.11	2.68	1.30	**0.25**	0.64
ED-10-07	47	6.18	2.87	0.79	2.11	**0.73**	5.69	2.79	**0.77**	2.33	1.12	5.70	2.79	1.49	0.82	**0.46**
ED-10-29	47	6.54	2.94	1.16	**0.55**	0.67	5.51	2.98	1.59	**0.38**	1.11	5.59	2.79	0.89	0.53	**0.46**
ED-10-18	49	6.84	3.20	1.46	**0.88**	1.50	6.68	3.28	1.33	**0.33**	1.43	6.35	3.26	1.41	**0.47**	0.89
ED-10-11	50	6.99	3.46	**0.67**	2.54	0.92	6.81	3.49	**1.01**	2.50	1.21	6.91	3.53	1.39	**0.68**	0.87
ED-10-02	51	7.54	3.78	0.55	2.85	**0.54**	8.22	4.09	**0.69**	1.74	1.02	8.13	4.26	1.95	2.23	**1.12**
ED-10-05	52	8.51	3.93	1.22	3.05	**0.97**	7.67	3.89	**0.70**	1.13	0.79	7.53	3.99	1.13	1.09	**0.98**
ED-10-15	52	8.40	3.94	**1.32**	2.23	1.68	7.69	3.88	**1.02**	2.31	1.45	7.65	3.93	1.98	2.23	**1.71**
ED-10-01	54	10.10	4.46	1.85	1.81	**0.89**	8.51	4.58	1.13	1.80	**1.00**	8.75	4.43	1.91	1.77	**1.18**
ED-10-03	54	10.16	5.13	**0.99**	2.59	1.99	9.40	4.26	**0.83**	2.41	1.06	8.70	4.42	1.53	1.50	**1.37**
MD-03-02	56	9.40	4.74	**0.66**	4.74	**0.66**	9.58	4.78	**0.41**	4.78	**0.41**	9.58	4.80	**0.46**	4.80	**0.46**
ED-10-16	57	11.26	5.49	1.74	**1.24**	1.61	10.00	5.42	**0.91**	1.32	1.20	10.46	5.48	1.68	**0.84**	0.95
MD-03-01	61	14.65	6.52	**0.42**	6.52	**0.42**	12.33	6.66	**0.47**	6.66	**0.47**	12.70	6.62	**0.37**	6.62	**0.37**
ED-10-17	61	13.48	6.48	1.27	5.24	**0.80**	12.70	6.37	**1.35**	1.61	1.76	12.49	6.32	1.38	1.54	**1.06**
ED-10-14	62	14.71	6.89	0.78	2.80	**0.67**	13.45	6.93	1.07	2.92	**0.74**	13.41	6.86	**1.27**	1.56	1.32
MD-03-04	63	13.86	7.12	**0.40**	7.12	**0.40**	13.66	6.99	**0.47**	6.99	**0.47**	13.73	7.33	**0.50**	7.33	**0.50**
ED-14-02	100	60.06	30.46	**0.73**	30.46	**0.73**	59.47	30.04	**0.75**	30.04	**0.75**	60.01	29.77	**0.76**	29.77	**0.76**
ED-14-03	100	60.08	30.41	**0.65**	30.40	**0.65**	59.42	29.96	**0.72**	29.96	**0.72**	60.09	29.53	**1.78**	29.53	**1.78**
MD-03-03	102	82.60	39.83	**0.59**	39.83	**0.59**	82.32	39.21	**0.69**	39.21	**0.69**	83.54	39.06	**1.02**	39.06	**1.02**
MD-03-05	103	84.11	38.35	**1.90**	38.35	**1.90**	81.71	38.35	**0.72**	38.35	**0.72**	81.48	38.92	**0.61**	38.92	**0.61**
MD-03-06	133	229.78	103.57	**1.18**	103.57	**1.18**	229.38	102.56	**1.26**	102.56	**1.26**	230.01	101.72	**1.20**	101.72	**1.20**
MD-03-08	147	305.83	136.26	**1.31**	136.26	**1.31**	307.80	136.98	**1.48**	136.98	**1.48**	303.46	137.33	**1.29**	137.33	**1.29**
MD-03-07	155	374.99	166.21	**1.64**	166.21	**1.64**	375.27	167.16	**1.57**	167.16	**1.57**	375.51	164.65	**1.46**	164.65	**1.46**
ED-10-50	170	1,144.02	207.24	152.53	202.63	**110.08**	$\geq 7,400.04^{\#}$	201.65	282.36	162.57	**159.5**	$\geq 7,392.48^{\&}$	251.31	**145.99**	252.3	192.23
ED-10-49	351	–	–	1393.73	–	**673.40**	–	–	4,260.8	–	**3,057.86**	–	–	5,956.47	–	**4,303.28**
Geometric Mean*		≥ 13.51	6.33	1.16	3.41	**1.06**	≥ 13.74	6.27	1.12	2.81	**1.09**	≥ 13.44	6.34	1.25	2.43	**0.90**

% The instance names have been shortened. The original names include three zeros before the firs number and six zeros before the second number
∗ The geometric mean does not include the ED-10-49 instance
The model had a relative optimality gap of 0.49% at the time of termination
& The model had a relative optimality gap of 0.12% at the time of termination

Table 4.2: Solution Time (in Seconds) of Different Exact Methods with and Without Prior GXCC Partitioning for Different Values of p for the Cohen-boulakia et al. Cohen-Boulakia *et al.* (2011) data set

Instance	n	$p = 1/2$					$p = 3/4$					$p = 1$				
		BBP	MIP	CR	GXCC_MIP	GXCC_CR	BBP	MIP	CR	GXCC_MIP	GXCC_CR	BBP	MIP	CR	GXCC_MIP	GXCC_CR
LQTS	35	2.19	1.11	1.11	**0.17**	0.86	2.20	1.05	1.25	**0.23**	0.43	2.50	1.15	1.26	**0.18**	0.25
ADHD	45	5.73	2.52	1.27	0.48	**0.43**	6.11	2.70	2.09	**0.41**	0.56	5.92	2.61	1.18	**0.39**	0.42
Prostate Caner	218	$\geq 7,421^{*}$	1,147.64	290.29	387.15	**64.27**	3,115.36	1,231.01	167.46	436.89	**43.66**	$\geq 7,389^{\#}$	1,199.35	132.29	423.15	**40.62**
Bladder Caner	308	–	–	526.08	1,034.96	**95.54**	–	–	226.31	1,042.19	**108.72**	–	–	255.08	1,030.73	**122.69**
Breast Caner	386	–	–	**2,254.51**	–	2,545.23	–	–	4,863.50	–	**3,332.51**	–	–	1,275.95	–	**945.36**
Retinoblastoma	402	–		1,245.65	–	**652.19**	–	–	673.61	–	**466.96**	–	–	659.78	–	**547.62**
Neuroblastoma	431	–		1,502.60	–	**396.60**	–	–	1,098.71	–	**369.04**	–	–	730.46	–	**347.05**

∗ The model had a relative optimality gap of 32.33% at the time of termination
The model had a relative optimality gap of 98.25% at the time of termination

MIP and CR were able to solve these two cases in less than 1,200 and 291 seconds, respectively. Additionally, "Bladder Cancer", "Breast Cancer", "Retinoblastoma", and "Neuroblastoma" could not be directly solved via BBP and MIP due to out-of-

Table 4.3: Number of Items in the Enlarged GXCC Partition's Subsets for Certain Large Instances

Instance	n	$\lvert X_1\rvert, \lvert X_2\rvert, \ldots, \lvert X_w\rvert$
ED-10-50	170	1, 5, 1, 1, 1, 161
ED-10-49	351	5, 3, 3, 7, 333
LQTS	35	3, 1, 1, 1, 3, 1, 2, 1, 2, 2, 16, 2
ADHD	45	1, 2, 5, 1, 3, 1, 1, 1, 1, 25, 4
Prostate Cancer	218	1, 17, 1, 166, 15, 16, 2
Bladder Cancer	308	1, 4, 21, 13, 3, 69, 197
Breast Cancer	386	1, 362, 11, 12
Retinoblastoma	402	1, 1, 1, 33, 1, 1, 2, 4, 17, 341
Neuroblastoma	431	6, 55, 29, 9, 9, 1, 322

memory errors. On the other hand, CR was able to solve each of these instances in less than 4,900 seconds. Interestingly, all instances of the TOC data set with 100 to 155 items did not require any of the preference-transitivity constraints to be included in the optimization model, which resulted in a significant differences in the run time of MIP and CR on those instances. As a final note, the average and maximum percent of preference-transitivity constraints added by the CR method were 0.61% and 5.41% for the TOC data set, and they were 2.67% and 7.48% for the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set.

Next, we examine the impact of enlarged GXCC partitioning on the run times of MIP and CR. Beforehand, Table 4.3 reports the size of subsets of the enlarged GXCC partitions for the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set and the two largest instances of the TOC data set; the partitions matched for each of tested values of p; other instances of TOC data set with more than 100 items were not partitionable. As Table 4.1 shows, enlarged GXCC partitioning was able to reduce the run times of both methods on the TOC data set for each of tested values of p. Impressively, it reduced the geometric mean run time of MIP from 6.34

to 2.43 seconds for $p = 1$. Enlarged GXCC partitioning decreased the run times of all instances with more than 62 items; however, it increased the run time of a handful of smaller instances. In fact, enlarged GXCC partitioning was able to reduce the run time of CR on "ED-10-49" approximately from 5,956 to 4,303 seconds, while it required only .02 seconds to obtain the partition. As Table 4.2 shows, enlarged GXCC partitioning reduced the run times of both methods on all instance of the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set for each of tested values of p. It is worth adding that MIP was not able to solve "Bladder Cancer" due to out-of-memory error; however, with the help of enlarged GXCC partitioning, MIP was able to solve this instance to optimality in approximately 1,043 seconds. Most impressively, enlarged GXCC partitioning was able to reduce the run time of CR on "Bladder Cancer" approximately from 526 to 95 seconds, a 5.5x improvement, and the run time of MIP on "Prostate Cancer" approximately from 1,098 to 369 seconds, close to a 3x improvement. The highest partitioning time of instances in this data set only took .07 seconds.

Next, we evaluate the performance of the LICH method. Table 4.4 reports the run time and %Deviation of the TOC data set; the best %Deviation attained for each instance and each of tested values of p is shown in bold. LICH achieved an average %Deviation of at most 0.90 and an geometric mean run time of 0.02 seconds on this data set. It obtained the optimal solution in 10 instances for each of tested values of p; its highest %Deviation on this data set was 3.29. Table 4.5 reports the run time and %Deviation of the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set; the best %Deviation attained for each instance and each of tested values of p is shown in bold. LICH achieved an average %Deviation of 1.93, 2.28, and 11.77 for $p = 1/2$, $p = 3/4$, and $p = 1$, respectively; its highest %Deviation was 37.50. However, the geometric mean run time of this method was less than one second on this data set,

Table 4.4: Solution Time (in Seconds) and %deviation of LICH with and Without Prior GXCC Partitioning for Different Values of p for TOC Instances With $40 \leq n \leq 400$

Instance	n	$p = 1/2$				$p = 3/4$				$p = 1$			
		Time		%Deviation		Time		%Deviation		Time		%Deviation	
		LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH
ED-10-21	40	0.01	0.01	**0.57**	0.85	0.01	0.01	0.39	**0.18**	0.01	0.01	0.38	**0.17**
ED-10-22	40	0.01	0.01	**0.92**	**0.92**	0.01	0.01	**0.70**	**0.70**	0.01	0.01	**0.71**	**0.71**
ED-10-30	40	0.01	0.01	0.68	**0**	0.01	0.01	0.35	**0**	0.01	0.01	0.62	**0.08**
ED-10-20	41	0.01	0.01	**0.30**	0.54	0.01	0.01	0.43	**0.27**	0.01	0.01	0.74	**0.54**
ED-10-31	41	0.01	0.01	1.35	**0.15**	0.01	0.01	0.67	**0.23**	0.01	0.01	0.82	**0.14**
ED-10-04	42	0.01	0.01	1.35	**1.15**	0.01	0.01	1.49	**0.48**	0.01	0.01	0.96	**0.82**
ED-10-09	42	0.01	0.01	2.75	**2.62**	0.01	0.01	**1.06**	**1.06**	0.01	0.01	**0.72**	**0.72**
ED-10-06	43	0.01	0.01	**0.45**	**0.45**	0.01	0.01	**0.34**	**0.34**	0.01	0.01	**0.29**	**0.29**
ED-10-10	43	0.01	0.01	**2.71**	**2.71**	0.01	0.01	**0.22**	**0.22**	0.01	0.01	**0.19**	0.21
ED-10-08	44	0.01	0.01	2.61	**2.11**	0.01	0.01	0.63	**0.09**	0.01	0.01	0.54	**0.08**
ED-10-12	44	0.01	0.01	1.48	**1.21**	0.01	0.01	1.14	**0.24**	0.01	0.01	1.30	**0.35**
ED-10-13	44	0.01	0.01	3.29	**0.12**	0.01	0.01	0.04	**0.01**	0.01	0.01	0.03	**0**
ED-10-34	46	0.01	0.01	0.50	**0.44**	0.01	0.01	0.36	**0.25**	0.01	0.01	0.43	**0.23**
ED-10-07	47	0.01	0.01	**1.06**	**1.06**	0.01	0.01	**0.42**	**0.42**	0.01	0.01	0.35	**0.15**
ED-10-29	47	0.01	0.01	0.75	**0.21**	0.01	0.01	0.56	**0.04**	0.01	0.01	0.81	**0**
ED-10-18	49	0.01	0.01	0.42	**0.32**	0.01	0.01	0.43	**0.06**	0.01	0.01	0.38	**0.05**
ED-10-11	50	0.01	0.01	**0.38**	0.40	0.01	0.01	**0.26**	**0.26**	0.01	0.01	0.34	**0.25**
ED-10-02	51	0.01	0.01	**0.55**	**0.55**	0.01	0.01	**1.47**	**1.47**	0.01	0.01	**.03**	**0.03**
ED-10-05	52	0.01	0.01	0.76	**0.49**	0.01	0.01	0.23	**0**	0.01	0.01	**0.14**	**0.14**
ED-10-15	52	0.01	0.01	**1.47**	**1.47**	0.01	0.01	1.02	**0.33**	0.01	0.01	0.83	**0.71**
ED-10-01	54	0.01	0.01	1.12	**0.51**	0.01	0.01	**1.60**	**1.60**	0.01	0.01	0.05	**0**
ED-10-03	54	0.01	0.01	**0.15**	**0.15**	0.01	0.01	0.29	**0.11**	0.01	0.01	0.30	**0.15**
MD-03-02	56	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**
ED-10-16	57	0.01	0.01	**1.12**	1.72	0.01	0.01	0.14	**0.05**	0.01	0.01	0.20	**0**
MD-03-01	61	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**
ED-10-17	61	0.01	0.01	**1.97**	**1.97**	0.01	0.01	0.53	**0.43**	0.01	0.01	0.17	**0.07**
ED-10-14	62	0.01	0.01	**0.97**	1.55	0.01	0.01	0.18	**0.06**	0.01	0.01	0.15	**0**
MD-03-04	63	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**	0.01	0.01	**0**	**0**
ED-14-02	100	0.19	0.19	**0**	**0**	0.19	0.19	**0**	**0**	0.23	0.23	**0**	**0**
ED-14-03	100	0.20	0.20	**0**	**0**	0.18	0.18	**0**	**0**	0.23	0.23	**0**	**0**
MD-03-03	102	0.19	0.19	**0**	**0**	0.19	0.19	**0**	**0**	0.24	0.24	**0**	**0**
MD-03-05	103	0.20	0.20	**0**	**0**	0.20	0.20	**0**	**0**	0.25	0.25	**0**	**0**
MD-03-06	133	0.46	0.46	**0**	**0**	0.44	0.44	**0**	**0**	0.55	0.55	**0**	**0**
MD-03-08	147	0.60	0.60	**0**	**0**	0.60	0.60	**0**	**0**	0.60	0.60	**0**	**0**
MD-03-07	155	0.67	0.67	**0**	**0**	0.72	0.72	**0**	**0**	0.85	0.85	**0**	**0**
ED-10-50	170	0.12	0.14	2.52	**2.01**	0.09	0.12	**1.31**	1.32	0.09	0.11	1.41	**1.21**
ED-10-49	351	2.33	2.27	1.00	**1.02**	1.22	1.19	**1.35**	1.39	1.52	1.52	**0.87**	0.93
Average		0.02	0.02	0.90	**0.78**	0.02	0.02	0.48	**0.31**	0.02	0.02	0.37	**0.22**

Table 4.5: Solution Time (in Seconds) and %deviation of LICH with and Without Prior GXCC Partitioning for Different Values of p of The Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set

Instance	n	$p = 1/2$				$p = 3/4$				$p = 1$			
		Time		%Deviation		Time		%Deviation		Time		%Deviation	
		LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH	LICH	GXCC_LICH
Long QT Syndrome	35	0.02	0.01	**0**	**0**	0.01	0.01	**0**	**0**	0.01	0.01	**3.69**	**3.69**
ADHD	45	0.01	0.01	**0**	**0**	0.01	0.01	6.74	**3.37**	0.01	0.01	**7.33**	**7.33**
Prostate Cancer	218	1.11	0.62	2.90	**2.85**	0.81	0.52	0.53	**0.51**	0.56	0.64	8.79	**0.35**
Bladder Cancer	308	2.11	0.40	0.41	**0.34**	1.99	1.46	1.09	**0.96**	2.46	1.76	0.12	**0.06**
Breast Cancer	386	5.91	4.97	**0.98**	**0.98**	4.89	3.89	**1.32**	**1.32**	2.28	1.89	37.50	**8.85**
Retinoblastoma	402	4.94	4.99	0.77	**0.05**	4.08	3.84	**0.28**	**0.28**	5.17	4.85	**0.54**	**0.54**
Neuroblastoma	431	4.59	4.58	9.30	**3.08**	5.21	4.63	4.06	**3.76**	1.29	0.41	15.83	**15.57**
Average		0.67	0.43	1.93	**1.19**	0.56	0.47	2.28	**1.66**	0.41	0.33	11.77	**5.20**

and its highest run time was 5.91 seconds.

Finally, we investigate the effect of enlarged GXCC partitioning on the run time and solution quality of LICH. As Table 4.4 shows, enlarged GXCC partitioning was able to slightly reduce the average %Deviation of the TOC data set for each of tested values of p, while maintaining the same geometric mean run time. On the other hand, it was able to reduce both the geometric mean run time and the average %Deviation of the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set for all of tested values of p, especially for $p = 1$. Remarkably, it reduced %Deviation of "Breast Cancer" for $p = 1$ from 37.50 to 8.85, and %Deviation of "Prostate Cancer" for $p = 1$ from 8.79 to 0.35.

Putting together all of these pieces, CR outperformed MIP, and MIP outperformed BBP. Additionally, enlarged GXCC partitioning reduced the run time of exact methods by up to 20x. The majority of the best run times of the exact methods were achieved by GXCC_CR. LICH achieved a near optimal solution on the most instances of the TOC data set, but it had a less commanding performance on the Cohen-Boulakia et al. Cohen-Boulakia *et al.* (2011) data set. However, enlarged GXCC partitioning reduced the run time and %Deviation of this method. Combining LICH with enlarged GXCC partitioning was shown to yield high-quality solutions in a short amount of time.

4.7 Conclusion

This chapter introduces and studies RANK-AGG, which contains KEMENY-AGG as a special case. It provides various analytical and computational contributions evaluated over two real-world data sets. It introduces a new mixed-integer programming formulation that outperformed a (revised) existing formulation over the featured instances. Additionally, it proposes a constraint relaxation technique, which was the

only exact method capable of solving several large instances (with up to 431 items). Furthermore, it presents a greedy heuristic algorithm for obtaining high-quality solutions to RANK-AGG. The overage %Deviation from optimality of this heuristic was 0.57 and 4.2 on the two tested data sets.

Additionally, this chapter broadens the applicability of Condorcet criterion variants to RANK-AGG by introducing a new social choice property (GXCC). It provides an algorithm for obtaining a valid GXCC partition and various analytical insights on the effect of the penalty parameter of the generalized Kendall-tau distance on the optimal ranking and GXCC partitions. GXCC proved to be effective in accelerating the run time of exact methods, as demonstrated by the featured experiments. It was able to decrease the run time exact and heuristic methods by up to 20x, and it improved %Deviation of the proposed heuristic by up to 19.14% percentage points.

Chapter 5

TOP-K LIST AGGREGATION

This chapter is organized as follows. Section 5.1 introduces the notations used throughout the chapter. Section 5.2 provides an overview of the distance-based top-k list aggregation problem. Section 5.3 introduces a binary nonlinear programming formulation and four mixed-integer linear programming (MIP) formulations of Top-k-Agg under the generalized Kendall-tau distance. Two of these formulations result from the introduction of preference cycle-prevention constraints specific to Top-k-Agg. Section 5.4 compares the strengths of the MIP formulations using techniques from polyhedral theory. Section 5.5 introduces various heuristic methods for solving the problem at hand. Section 5.6 proposes a data reduction technique to accelerate the solution to certain large-scale instances. Section 5.7 conducts computational experiments and reports their results. Section 5.8 studies the university rankings aggregation problem as a case study. Finally, Section 5.9 concludes the chapter.

5.1 Notations

Let n denote the number of items and $\boldsymbol{\mathcal{B}}$ denote an ordering of n items. A top-k list $\boldsymbol{\tau}$ is a bijection from a domain $\boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}$ (the members of $\boldsymbol{\tau}$) to $[k] = \{1, \ldots, k\}$, where $k < n$. All items in $\boldsymbol{\tau}$ are presumed to be ranked ahead of items not in $\boldsymbol{\tau}$. The exact ordering of items not in the list is unknown; however, it is implicitly assumed that all of these items are tied at position $k + 1$. Let $i \in \boldsymbol{\tau}$ indicate that item i appears in the top-k list, and let $\boldsymbol{\tau}(i)$ denote the rank or position of i therein. Additionally, let $i \succ_{\boldsymbol{\tau}} j$ denote that item i is rank ahead of item j in $\boldsymbol{\tau}$, that is if i but not j is present in $\boldsymbol{\tau}$ or if both are present and i is ranked ahead of j, that is $(i \in \boldsymbol{\tau} \wedge j \notin \boldsymbol{\tau})$

OR $(i, j \in \boldsymbol{\tau} \wedge (\boldsymbol{\tau}(i) < \boldsymbol{\tau}(j)))$. Given top-$k$ lists $\boldsymbol{\tau}^1$ and $\boldsymbol{\tau}^2$, let $\boldsymbol{\Lambda}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ be the set of all unordered pairs of distinct items in $\boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}^1} \bigcup \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}^2}$.

5.2 Distance-based Top-k List Aggregation

This section reviews existing distance-based top-k list aggregation techniques. As mentioned in Section 1, these techniques can be categorized based on whether the output ranking is considered a full list over $\boldsymbol{\mathcal{I}}$ or another top-k list. Note, however, that if a smaller consensus top-k list is desired in the former category, it can be extracted from the full consensus list. Dwork *et al.* (2001), Ailon (2010), and Nápoles *et al.* (2017) fall into the first category; Fagin *et al.* (2003) falls into the second category. The works referenced under the first category define the top-k list aggregation problem as finding a complete list with the least cumulative distance to the input lists using the induced Kendall-tau, Kendall-tau, and Hausdorff distances, respectively. The ensuing paragraphs explore two Kendall-tau distance-based variants.

Dwork *et al.* (2001) introduced the top-k list aggregation problem under the induced Kendall-tau distance, where the goal is to find a full list with the least cumulative such distance to the input lists. Given a full list $\boldsymbol{\mathcal{B}}$ and a top-k list $\boldsymbol{\tau}$, the projection of $\boldsymbol{\mathcal{B}}$ with respect to $\boldsymbol{\tau}$, denoted as $\boldsymbol{\mathcal{B}}_{|\boldsymbol{\tau}}$, is a new reduced top-$k$ list that contains only those elements in $\boldsymbol{\tau}$. The induced Kendall-tau distance then is obtained by calculating the Kendall-tau distance between the projection of the lists with respect to the set of items they share in common (i.e., since the projected lists are equal in size). For example, let $\boldsymbol{\mathcal{B}} = \{1, 2, 3, 4, 5, 6\}$ be an ordering of 6 items, and $\boldsymbol{\tau} = \{5, 3, 2\}$ be a top-3 list. Here, $\boldsymbol{\mathcal{B}}_{|\boldsymbol{\tau}} = \{2, 3, 5\}$ and the induced Kendall-tau distance between $\boldsymbol{\mathcal{B}}_{|\boldsymbol{\tau}}$ and $\boldsymbol{\tau}$ is 3. Let $\boldsymbol{\mathcal{L}} = \{1, 2, \ldots, m\}$ be the set of indices of the input top-k lists and $\boldsymbol{\tau}^l$ be the input top-k list $l \in \boldsymbol{\mathcal{L}}$. The optimal solution to this

problem can be written as

$$\boldsymbol{\mathcal{B}}^* = \underset{\boldsymbol{\mathcal{B}} \in \boldsymbol{\Sigma}}{\arg\min} \sum_{l \in \boldsymbol{\mathcal{L}}} K(\boldsymbol{\mathcal{B}}_{|\boldsymbol{\tau}^l}, \boldsymbol{\tau}^l). \tag{5.1}$$

Ailon (2010) proposed an alternative variant, where the goal is to find a full list of items with the least cumulative Kendall-tau distance to the input lists. The Kendall-tau distance between a full list and a top-k list is equal to the number of item-pairs whose relative orderings are different. The optimal solution to this problem can be written as

$$\boldsymbol{\mathcal{B}}^* = \underset{\boldsymbol{\mathcal{B}} \in \boldsymbol{\Sigma}}{\arg\min} \sum_{l \in \boldsymbol{\mathcal{L}}} K(\boldsymbol{\mathcal{B}}, \boldsymbol{\tau}^l). \tag{5.2}$$

Dwork *et al.* (2001)'s method retains less information compared to Ailon (2010)'s method. Expressly, Ailon (2010)'s method uses the information that items not on a top-k list are ranked lower than all items on the list, whereas the projection operation completely discards this information. Nonetheless, both approaches discard the partial information corresponding to case where two items are present in one list but neither is present in the other. Additionally, neither of these variants is a metric nor a pseudo-metric (Ailon, 2010; Dwork *et al.*, 2001), meaning that they do not satisfy the triangle inequality—a fundamental property of distance measures (Brandt *et al.*, 2016)—or even a relaxed version of it. This also means that the consensus top-k lists obtained by these approaches may not match.

Among the aforementioned distance-based approaches, Fagin *et al.* (2003)'s method provides higher flexibility and it induces a far smaller solution space. Specifically, since the output is another top-k list rather than a full list over $\boldsymbol{\mathcal{I}}$, there are $|\boldsymbol{\mathcal{T}}| = \binom{n}{k} k!$ possible top-k lists, which is $(n-k)!$ times smaller than $n!$ (the number of possible complete strict lists over $\boldsymbol{\mathcal{I}}$). As such, this method provides relatively more flexibility and practicality. Next, we formally present the associated top-k list aggregation

problem, which is denoted as TOP-k-AGG.

Definition 17 *(TOP-k-AGG) Let $\boldsymbol{\mathcal{L}} = \{1, 2, \ldots, m\}$ be the set of indices of the input top-k lists, $\boldsymbol{\tau}^l$ be the input top-k list $l \in \boldsymbol{\mathcal{L}}$, $\boldsymbol{\mathcal{I}} = \bigcup_{l \in \boldsymbol{\mathcal{L}}} \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}^l}$ be the universe of items, $n := |\boldsymbol{\mathcal{I}}|$ be the number of items in the universe $\boldsymbol{\mathcal{I}}$, $\boldsymbol{\mathcal{T}}$ be the set of all possible top-k lists over $\boldsymbol{\mathcal{I}}$, and $d(.,.)$ be a distance measure between top-k lists. TOP-k-AGG seeks to find a top-k list $\boldsymbol{\tau}^* \in \boldsymbol{\mathcal{T}}$ with the lowest cumulative distance to the input lists; it can be written succinctly as*

$$\boldsymbol{\tau}^* = \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \sum_{l \in \boldsymbol{\mathcal{L}}} d(\boldsymbol{\tau}, \boldsymbol{\tau}^l). \tag{5.3}$$

Going forward, we refer to optimal solution of TOP-k-AGG as the *consensus top-k list*.

Various distance measures between top-k lists have been introduced including generalized Kendall-tau, generalized Spearman's footrule, Spearman's rho (Fagin *et al.*, 2003), Hausdorff, (Critchlow, 2012; Fagin *et al.*, 2003), Goodman and Kruskal's gamma (Goodman and Kruskal, 1959), and Canberra (Jurman *et al.*, 2009). Additionally, there are other ways to compare top-k lists. For example, Collier and Konagurthu (2014) defines an information-theoretic perspective and focuses on the amount of information conveyed. More specifically, this method compares top-k lists based on their compressibility, which is related to their similarity (i.e., lists with more information in common are more compressible). We refer the reader to Fagin *et al.* (2003) for more information on comparing top-k lists.

5.2.1 Generalized Kendall-tau Distance

The rest of this chapter focuses on the generalized Kendall-tau distance (Fagin *et al.*, 2003) since the Kendall-tau distance and its variants have been shown to be effective for rank aggregation (Dwork *et al.*, 2001), especially in the context of infor-

mation retrieval (Fagin *et al.*, 2003). Despite the focus on this particular distance, the mathematical formulations and polyhedral analyses presented herein can be extended to TOP-k-AGG under any other distance measure between top-k lists by modifying the objective functions accordingly. This distance measure is restated in the following. Let p be a fixed parameter, with $0 \leq p \leq 1$, and let $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ be the contribution to the distance function, for each item-pair $(i, j) \in \boldsymbol{\Lambda}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$. $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ is divided into four cases based on the relative positions of i and j within $\boldsymbol{\tau}^1$ and $\boldsymbol{\tau}^2$:

Case 1. i and j appear in both top-k lists. If i and j are in the same order in both lists, then set $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = 0$; otherwise, set $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = 1$.

Case 2. i and j appear in one top-k list, say $\boldsymbol{\tau}^1$, and exactly one of i or j, say i, appears in the other top-k list ($\boldsymbol{\tau}^2$) . If i is ranked ahead of j in $\boldsymbol{\tau}^1$, then set $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = 0$; otherwise, set $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = 1$.

Case 3. i, but not j, appears in one top-k list, say $\boldsymbol{\tau}^1$, and j but not i, appears in the other top-k list, $\boldsymbol{\tau}^2$; then, set $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = 1$.

Case 4. i and j both appear in one top-k list, say $\boldsymbol{\tau}^1$, but neither i nor j appear in the other list ($\boldsymbol{\tau}^2$). Such pairs are called *special pairs*, and the penalty is set to the fixed penalty parameter, that is, $K_{i,j}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = p$.

Piecing together the above four cases, $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ can be succinctly written as

$$K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = \begin{cases} 1 & (i \succ_{\boldsymbol{\tau}^1} j \ \wedge \ j \succ_{\boldsymbol{\tau}^2} i) \ \vee \ (j \succ_{\boldsymbol{\tau}^2} i \ \wedge \ i \succ_{\boldsymbol{\tau}^1} j) \\ p & (i, j \in \boldsymbol{\tau}^1 \ \wedge \ i, j \notin \boldsymbol{\tau}^2) \ \vee \ (i, j \notin \boldsymbol{\tau}^1 \ \wedge \ i, j \in \boldsymbol{\tau}^2) \\ 0 & \text{otherwise.} \end{cases}$$

Considering all distinct item-pairs, the *generalized Kendall-tau distance with penalty parameter* p, denoted as $K^{(p)}$, is defined as

$$K^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2) = \sum_{(i,j) \in \boldsymbol{\Lambda}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)} K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2). \tag{5.4}$$

$K^{(p)}$ is a *near metric* since it satisfies a relaxed version of the triangle inequality (Fagin *et al.*, 2003). TOP-k-AGG under $K^{(p)}$ is a combinatorial NP-hard problem (Fagin *et al.*, 2003), which includes KEMENY-AGG with strict rankings as a special case (when $k = n$).

In analogous fashion, Fagin *et al.* (2003) proposed the *footrule distance with location parameter* l, denoted by $F^{(l)}$, to generalize the footrule distance for comparing top-k lists. Distance $F^{(l)}$ is obtained by placing all items not in a top-k list at position l (this transforms a top-k list into a full list) and then computing the footrule distance of the completed lists. An intuitive and convenient choice is to set $l = k + 1$.

Chierichetti *et al.* (2018) generalized Mallow's model to distribution on top-k lists using $K^{(p)}$. The model induces a distribution on $\boldsymbol{\mathcal{T}}$ such that

$$Pr[\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}] = \frac{1}{Z_\beta} \exp(-\beta . K^{(p)}(\boldsymbol{\tau}', \boldsymbol{\tau})), \tag{5.5}$$

where β is the decay parameter, Z_β is a normalizing constant, and $\boldsymbol{\tau}'$ is the ground-truth top-k list. Similar to full strict rankings, the solution to TOP-k-AGG can be interpreted statistically as a maximum likelihood estimator wherein the input lists are independent realizations of the Mallows probabilistic model using distance $K^{(p)}$, that is,

$$\boldsymbol{\tau}^* = \underset{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}}{\operatorname{argmin}} \sum_{l \in \boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\tau}, \boldsymbol{\tau}^l) = \underset{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}}{\operatorname{argmax}} \, \Pi_{l \in \boldsymbol{\mathcal{L}}} Pr[\boldsymbol{\tau}^l].$$

5.3 Exact Optimization Models and Theoretical Insights

To the best of our knowledge, no efforts have been made to derive an explicit mathematical model of TOP-k-AGG. This section presents various formulations.

First, we define the required parameters for defining the objective functions of the presented formulations of TOP-k-AGG under $K^{(p)}$. Let μ_{il} be an indicator parameter that is equal to 1 if $i \in \boldsymbol{\tau}^l$, where $l \in \boldsymbol{\mathcal{L}}$. Additionally, let s_{ij} denote the number of

input lists where item i is ranked ahead of item j, which can be expressed as

$$\begin{aligned} s_{ij} &= \sum_{l \in \boldsymbol{\mathcal{L}}} \mathbb{1}_{(i,j \in \boldsymbol{\tau}^l \wedge (\boldsymbol{\tau}^l(i) < \boldsymbol{\tau}^l(j)) \vee (i \in \boldsymbol{\tau}^l \wedge j \notin \boldsymbol{\tau}^l)} \\ &= \sum_{l \in \boldsymbol{\mathcal{L}}} \left[\mu_{il}\, \mu_{jl}\, \mathbb{1}_{\boldsymbol{\tau}^l(i) < \boldsymbol{\tau}^l(j)} + \mu_{il}(1 - \mu_{jl}) \right]. \end{aligned} \tag{5.6}$$

In words, s_{ij} tallies the number of input lists in which i is ranked ahead of j, that is, the number of input lists in which both items are present and i is ranked ahead of j, plus the number of inputs lists in which i is present but j is not.

Using these parameters, the cumulative $K^{(p)}$ distance between a given top-k list $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ and all of the input top-k lists, i.e., $\sum_{\boldsymbol{\tau}^l \in \boldsymbol{\mathcal{L}}} \sum_{(i,j) \in \boldsymbol{\Lambda}(\boldsymbol{\tau}, \boldsymbol{\tau}^l)} K_{ij}^{(p)}(\boldsymbol{\tau}, \boldsymbol{\tau}^l)$, can be expressed as $\sum_{(i,j) \in \boldsymbol{\Lambda}} K_{ij}^{(p)}(\boldsymbol{\tau})$ where $\boldsymbol{\Lambda}$ is set of all unordered pairs of distinct items in $\boldsymbol{\mathcal{I}}$, and

$$K_{ij}^{(p)}(\boldsymbol{\tau}) = \begin{cases} s_{ji} + p \sum_{l \in \boldsymbol{\mathcal{L}}} (1 - \mu_{il})(1 - \mu_{jl}) & \text{if } i, j \in \boldsymbol{\tau} \wedge (\boldsymbol{\tau}(i) < \boldsymbol{\tau}(j)), \\ s_{ji} & \text{if } i \in \boldsymbol{\tau} \wedge j \notin \boldsymbol{\tau}, \\ p \sum_{l \in \boldsymbol{\mathcal{L}}} \mu_{il} \mu_{jl} & \text{if } i, j \notin \boldsymbol{\tau}. \end{cases} \tag{5.7}$$

Eq. (5.7) states that whenever item i and j are both present in $\boldsymbol{\tau}$ (the solution top-k list) and i is ranked ahead of item j, the imposed $K^{(p)}$ distance between $\boldsymbol{\tau}$ and all of the input lists for this pair of items equals the number of input lists where j is ranked ahead of i, plus p-times the number of input lists neither i nor j is present in the same list. Whenever i but not j is present in $\boldsymbol{\tau}$, the imposed $K^{(p)}$ distance equals the number of input lists where j is ranked ahead of i. Finally, whenever neither i nor j is present in $\boldsymbol{\tau}$, the imposed $K^{(p)}$ distance equals p times the number of input lists where i and j are simultaneously present.

The first formulation is a MIP possessing an assignment problem-like structure, with which exactly k items are assigned to the k available positions of the solution

top-k list. Its decisions variables are as follows:

$$u_{it} = \begin{cases} 1 & \text{if } i \text{ is assigned to position } t \in [k] \\ 0 & \text{otherwise;} \end{cases}$$

$$w_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are in the top-}k \text{ list, and } i \text{ is ranked ahead of } j \\ 0 & \text{otherwise;} \end{cases}$$

$$w'_{ij} = \begin{cases} 1 & \text{if } i \text{ is in the top-}k \text{ list, but not } j \\ 0 & \text{otherwise;} \end{cases}$$

$$w''_{ij} = \begin{cases} 1 & \text{if neither } i \text{ nor } j \text{ is present in the top-}k \text{ list, where } j > i \\ 0 & \text{otherwise.} \end{cases}$$

From the definitions, item i is present in the top-k list if $\sum_{t=1}^{k} u_{it} = 1$, and it is absent if $\sum_{t=1}^{k} u_{it} = 0$. The variables $\boldsymbol{w}, \boldsymbol{w'}$, and $\boldsymbol{w''}$ determine the relative ordering of the items; these are dependent variables, as their exact values are determined by the values of the $\boldsymbol{u}$-variables. The first formulation (MIP#1) is as follows.

$$\min_{u,w,w',w''} \quad \sum_{i\in\boldsymbol{\mathcal{I}}}\sum_{j\in\boldsymbol{\mathcal{I}}}\left[(s_{ji} + p\sum_{l\in\boldsymbol{\mathcal{L}}}(1-\mu_{il})(1-\mu_{jl}))w_{ij} + s_{ji}w'_{ij}\right] + p\sum_{i,j\in\boldsymbol{\mathcal{I}},j>i}\sum_{l\in\boldsymbol{\mathcal{L}}}\mu_{il}\mu_{jl}w''_{ij} \tag{5.8a}$$

$$\text{s.t.} \quad \sum_{i\in\boldsymbol{\mathcal{I}}} u_{it} = 1 \qquad \forall t \in [k] \tag{5.8b}$$

$$\sum_{t\in[k]} u_{it} \leq 1 \qquad \forall i \in \boldsymbol{\mathcal{I}} \tag{5.8c}$$

$$w_{ij} \geq \sum_{t'=1}^{t} u_{it'} + \sum_{t''=t+1}^{k} u_{jt''} - 1 \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\ i \neq j;\ \forall t \in [k-1] \tag{5.8d}$$

$$\sum_{i,j\in\boldsymbol{\mathcal{I}}} w_{ij} \leq \frac{k(k-1)}{2} \tag{5.8e}$$

$$w'_{ij} \geq \sum_{t\in[k]} u_{it} - \sum_{t\in[k]} u_{jt} \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\, i \neq j \tag{5.8f}$$

$$\sum_{i,j\in\boldsymbol{\mathcal{I}}} w'_{ij} = k(n-k) \tag{5.8g}$$

$$w''_{ij} \geq 1 - \sum_{t\in[k]} u_{it} - \sum_{t\in[k]} u_{jt} \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\, i \neq j \tag{5.8h}$$

$$\sum_{i,j\in\boldsymbol{\mathcal{I}}, j>i} w''_{ij} = \frac{(n-k)(n-k-1)}{2} \tag{5.8i}$$

$$u_{it} \in \{0,1\} \qquad \forall i \in \boldsymbol{\mathcal{I}};\;\; \forall t \in [k] \tag{5.8j}$$

$$w_{ij}, w'_{ij} \geq 0 \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\, i \neq j \tag{5.8k}$$

$$w''_{ij} \geq 0 \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\;\; j > i. \tag{5.8l}$$

Objective function (5.8a) minimizes the cumulative $K^{(p)}$ distance to the input lists according to Eq. (5.7). Constraint (5.8b) enforces that exactly one item must be assigned to each position of the top-k list. Constraint (5.8c) enforces that every item must be assigned to at most one position of the list. Constraint (5.8d) determines the respective values of the $\boldsymbol{w}$-variables. More specifically, $w_{ij} = 1$ if i occupies one of the first t positions ($\sum_{t'=1}^{t} u_{it'} = 1$) and j occupies position t'', where $t+1 \leq t'' \leq k$ ($\sum_{t''=t+1}^{k} u_{jt''} = 1$); otherwise, this constraint becomes redundant. Constraint (5.8d) and (5.8e) together impose preference transitivity (i.e., prevent preference cycles); this means that if h is ranked ahead of i, and i is ranked of j, then h must be ranked ahead of j as well (see Theorem 10). Constraint (5.8f) determines the respective values of $\boldsymbol{w'}$-variables; it enforces that $w'_{ij} = 1$ if i is present in the top-k list but not j; otherwise, this constraint becomes redundant. Constraint (5.8g) enforces that at most $k(n-k)$ of the $\boldsymbol{w'}$-variables can take a value of 1 as there are $k(n-k)$ distinct item-pairs where exactly one of the items appears in the list. Constraint (5.8h) enforces that $w''_{ij} = 1$ if neither i nor j is present in the top-k list; otherwise, this constraint becomes redundant. Constraint (5.8i) enforces that at most $(n-k)(n-k-1)/2$ of the $\boldsymbol{w''}$-

variables can take a value of 1 as this is the number of distinct item-pairs where both items are absent from the list. Constraints (5.8j)-(5.8l) specify the domain of the variables.

Taking a closer look at the structure of the constraints, we can observe that even though variables $\boldsymbol{w}, \boldsymbol{w'}$ and $\boldsymbol{w''}$ are specified as binary, they can be treated as non-negative continuous variables since the constraints of the model alone enforce them to only take a value of 0 or 1. It is important also to remark that the reason for including Constraints (5.8f) and (5.8g) is that the objective function coefficients are not necessarily positive. More specifically, if both i and j are present in the solution top-k list, Constraint (5.8f) implies that $w'_{ij} \geq 0$; however, if the objective function coefficient s_{ij} is 0, then any value of w'_{ij} results in the same objective function value, which is not desirable.

Theorem 10 *Constraints* (5.8d)-(5.8e) *impose preference transitivity.*

Assume that items h, i, j are present in the solution top-k list with h placed in position $t \geq 1$, i in position $t' > t$, and j in position t'', where $k \geq t'' > t'$. Constraint (5.8d) enforces that $w_{hi} = w_{hj} = w_{ij} = 1$. However, this constraint only implies that $w_{jh} \geq -1$. In other words, the optimization model may have an incentive to assign $w_{jh} = 1$, creating a preference cycle, in order to decrease the objective function value. Hence, Constraint (5.8d) on its own does not prevent preference cycles.

However, the total number of $\boldsymbol{w}$-variables that *must* take a value of 1 is given by $(k-1) + (k-2) + \cdots + 1 + 0 = k(k-1)/2$—the first-ranked item is ahead of $k-1$ other items in the list, the second-ranked item is ahead of $k-2$ items, ..., and the item at the bottom of the list is not ranked ahead of any other items on the list. For this reason, Constraint (5.8e) allows at most $k(k-1)/2$ of the $\boldsymbol{w}$-variables to take a value of 1, forcing all other variables (including w_{jh}) to equal 0. Therefore,

Constraints (5.8d)-(5.8e) together impose preference transitivity on the solution top-k list returned by solving MIP#1.

Since KEMENY-AGG is a special case of TOP-k-AGG, MIP#1 provides a novel formulation for that problem as well; however, it does not apply to the variant of the problem with ties. It is important to mention that Cook (2006) proposed a binary linear programming formulation of KEMENY-AGG using the structure of the assignment problem; however, their set of preference cycle prevention constraint is different from Constraints (5.8d)-(5.8e).

Next, we present a binary non-linear programming formulation for TOP-k-AGG. The formulation uses the $\boldsymbol{w}$-variables defined for MIP#1 as well as the following decision variables:

$$z_i = \begin{cases} 1 & \text{if } i \text{ is in the top-}k \text{ list} \\ 0 & \text{otherwise.} \end{cases}$$

The formulation is given by:

$$\min_{\boldsymbol{w},\boldsymbol{z}} \quad \sum_{i\in\boldsymbol{\mathcal{I}}}\sum_{j\in\boldsymbol{\mathcal{I}}}\left[(s_{ji} + p\sum_{l\in\boldsymbol{\mathcal{L}}}(1-\mu_{il})(1-\mu_{jl}))w_{ij} + s_{ji}z_i(1-z_j)\right] + p\sum_{i,j\in\boldsymbol{\mathcal{I}},j>i}\sum_{l\in\boldsymbol{\mathcal{L}}}\mu_{il}\mu_{jl}(1-z_i)(1-z_j) \tag{5.9a}$$

$$\text{s.t.} \quad \sum_{i\in\boldsymbol{\mathcal{I}}} z_i = k \tag{5.9b}$$

$$w_{hi} + w_{ij} + w_{jh} \leq 2 \qquad \forall h,i,j \in \boldsymbol{\mathcal{I}},\ i,j > h,\ i \neq j \tag{5.9c}$$

$$w_{ij} + w_{ji} = z_i z_j \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\ j > i \tag{5.9d}$$

$$z_i, w_{ij} \in \{0,1\} \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\ i \neq j. \tag{5.9e}$$

Objective function (5.9a) minimizes the cumulative $K^{(p)}$ distance to the input lists. Constraint (5.9b) restricts k items to be present in the top-k list. Constraint (5.9c) imposes preference transitivity only whenever items h,i,j all appear in the list; oth-

erwise, it becomes redundant, with the help of Constraint (5.9d). Constraint (5.9d) enforces that, when both i and j are present in the list, one must proceed the other. Constraint (5.9e) specifies the domains of the variables. Given a feasible solution, the output top-k items are defined by the set $\overline{\boldsymbol{\tau}} := \{i \in \boldsymbol{\mathcal{I}} | z_i = 1\}$, and the exact rank of item $i \in \overline{\boldsymbol{\tau}}$ is obtained as $\overline{\boldsymbol{\tau}}(i) := k - \sum_{j \in \overline{\boldsymbol{\tau}}} w_{ij}$.

The above non-linear optimization model can be linearized using a technique from Glover and Woolsey (1974). Specifically, Constraint (5.9d) can be replaced with three linear constraints for each distinct item pair (i, j): $w_{ij} + w_{ji} \leq z_i$, $\quad w_{ij} + w_{ji} \leq z_j$, and $w_{ij} + w_{ji} \geq z_i + z_j - 1$. Similarly, the term $z_i(1 - z_j)$ in the objective function is replaced by auxiliary continuous variable x'_{ij} and constraints $x'_{ij} \geq z_i - z_j$ and $x'_{ij} \geq 0$; and the term $(1 - z_i)(1 - z_j)$ in the objective function is replaced by auxiliary continuous variable x''_{ij} and constraints $x''_{ij} \geq 1 - z_i - z_j$ and $x''_{ij} \geq 0$. The latter two cases use the fact the objective function coefficients of $z_i(1 - z_j)$ and $(1 - z_i)(1 - z_j)$ are non-negative, leading to a reduction in the number of constraints required by the linearization. The resulting formulation (MIP#2) is given by:

$$\min_{\boldsymbol{w}, \boldsymbol{x}', \boldsymbol{x}'', \boldsymbol{z}} \quad \sum_{i \in \boldsymbol{\mathcal{I}}} \sum_{j \in \boldsymbol{\mathcal{I}}} \left[(s_{ji} + p \sum_{l \in \boldsymbol{\mathcal{L}}} (1 - \mu_{il})(1 - \mu_{jl})) w_{ij} + s_{ji} x'_{ij} \right] + p \sum_{i,j \in \boldsymbol{\mathcal{I}}, j > i} \sum_{l \in \boldsymbol{\mathcal{L}}} \mu_{il} \mu_{jl} x''_{ij} \tag{5.10a}$$

$$\text{s.t.} \quad (5.9b), (5.9c), (5.9e) \tag{5.10b}$$

$$w_{ij} + w_{ji} \geq z_i + z_j - 1 \qquad \forall i, j \in \boldsymbol{\mathcal{I}},\ j > i \tag{5.10c}$$

$$w_{ij} + w_{ji} \leq z_i \qquad \forall i, j \in \boldsymbol{\mathcal{I}},\ i \neq j \tag{5.10d}$$

$$x'_{ij} \geq z_i - z_j \qquad \forall i, j \in \boldsymbol{\mathcal{I}},\ i \neq j \tag{5.10e}$$

$$\sum_{i,j \in \boldsymbol{\mathcal{I}}} x'_{ij} = k(n - k) \tag{5.10f}$$

$$x''_{ij} \geq 1 - z_i - z_j \qquad \forall i, j \in \boldsymbol{\mathcal{I}},\ j > i \tag{5.10g}$$

$$\sum_{i,j\in\boldsymbol{\mathcal{I}},j>i} x''_{ij} = \frac{(n-k)(n-k-1)}{2} \tag{5.10h}$$

$$x'_{ij} \geq 0 \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\, i \neq j, \tag{5.10i}$$

$$x''_{ij} \geq 0 \qquad \forall i,j \in \boldsymbol{\mathcal{I}},\, j > i. \tag{5.10j}$$

The rationale behind including Constraints (5.10f) and (5.10h) is the same as Constraints (5.8g) and (5.8i) in MIP#1.

Next, we define two variants of the preference transitivity constraints utilized in MIP#2.

Proposition 9 *Constraint* (5.9c) *can be replaced by non-linear constraints*

$$w_{hi} + w_{ij} + w_{jh} \leq 3 - z_h z_i z_j \qquad \forall i,j > h,\, i \neq j, \quad \textbf{\textit{or}} \tag{5.11}$$

$$w_{hi} + w_{ij} + w_{jh} \leq 1 + z_h z_i z_j \qquad \forall i,j > h,\, i \neq j. \tag{5.12}$$

Furthermore, these constraints can be linearized respectively as

$$w_{hi} + w_{ij} + w_{jh} \leq 3 - \frac{1}{3}(z_h + z_i + z_j) \quad \forall h,i,j \in \boldsymbol{\mathcal{I}},\, i,j > h,\, i \neq j, \tag{5.13}$$

$$w_{hi} + w_{ij} + w_{jh} \leq 1 + \frac{1}{3}(z_h + z_i + z_j) \quad \forall h,i,j \in \boldsymbol{\mathcal{I}},\, i,j > h,\, i \neq j. \tag{5.14}$$

The right-hand side of Constraints (5.11)-(5.14) becomes 2, as desired, when items h,i,j are all in the solution top-k list, i.e., when $z_h = z_i = z_j = 1$. For the remaining cases, these constraints become redundant, with the help of Constraint (5.10d). In particular, assume i is not in the top-k list; Constraint (5.10d) enforces that $w_{ij}+w_{ji} \leq 0$ and $w_{ih} + w_{hi} \leq 0$; hence, Constraints (5.11)-(5.14) effectively reduce to $w_{jh} \leq 1$, which is redundant. Replacing Constraint (5.9c) with Constraints (5.13) and (5.14), respectively, induces two additional MIPs.

MIP#3:

$$\min_{\boldsymbol{w},\boldsymbol{x'},\boldsymbol{x''},\boldsymbol{z}} \tag{5.10a}$$

$$\text{s.t.} \quad (5.9b), (5.9e), (5.10c) - (5.10g)$$
$$w_{hi} + w_{ij} + w_{jh} \leq 3 - \frac{1}{3}(z_h + z_i + z_j) \quad \forall h, i, j \in \boldsymbol{\mathcal{I}},\, i, j > h,\, i \neq j.$$

MIP#4:

$$\min_{\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z}} \quad (5.10a)$$
$$\text{s.t.} \quad (5.9b), (5.9e), (5.10c) - (5.10g)$$
$$w_{hi} + w_{ij} + w_{jh} \leq 1 + \frac{1}{3}(z_h + z_i + z_j) \quad \forall h, i, j \in \boldsymbol{\mathcal{I}},\, i, j > h,\, i \neq j.$$

5.3.1 Effect of the Penalty Parameter p on the Consensus List

Similar to RANK-AGG, the value of penalty parameter p could have a big impact on the resulting consensus top-k list. It is worth highlighting three settings of p previously discussed in Fagin *et al.* (2003). Using $p = 0$ is called the *optimistic approach*; it corresponds to the situation where $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ is set to 0 whenever there is not enough information to determine whether the penalty should be 0 or 1. Using $p = 0.5$ is called the *neutral approach*; it corresponds to the situation where $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ is set to 0.5 when there is not enough information to determine whether the penalty should be 0 or 1 (Fagin *et al.*, 2003). Conversely, using $p = 1$ can be called the *pessimistic approach*; it corresponds to the situation where $K_{ij}^{(p)}(\boldsymbol{\tau}^1, \boldsymbol{\tau}^2)$ is set to 1 whenever there is not enough information to determine whether the penalty should be 0 or 1. Beyond these general descriptions, there have been no attempts to help guide the choice of p or to analyze its implications. The ensuing paragraphs provide useful insights for this purpose.

Proposition 10 *The set of optimal objective values of problem* (5.3) *over all values of* $p \in [0, 1]$ *forms a piecewise linear envelope.*

The cumulative $K^{(p)}$ distance of a solution top-k list $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ to the input lists, $\sum_{l \in \boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\tau}, \boldsymbol{\tau}^l)$, can be expressed as $a^{(\boldsymbol{\tau})} + pb^{(\boldsymbol{\tau})}$, which is an affine function in terms of p, where

$$a^{(\boldsymbol{\tau})} = \sum_{i,j \in \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}} \left(s_{ji} \mathbb{1}_{\boldsymbol{\tau}(i) < \boldsymbol{\tau}(j)} + s_{ij} \mathbb{1}_{\boldsymbol{\tau}(j) < \boldsymbol{\tau}(i)} \right) + \sum_{i \in \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}} \sum_{j \notin \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}} s_{ji},$$

and

$$b^{(\boldsymbol{\tau})} = \sum_{i,j \in \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}} \sum_{l \in \boldsymbol{\mathcal{L}}} (1 - \mu_{il})(1 - \mu_{jl}) + \sum_{i,j \in \boldsymbol{\mathcal{I}} \setminus \boldsymbol{\mathcal{I}}_{\boldsymbol{\tau}}} \sum_{l \in \boldsymbol{\mathcal{L}}} \mu_{il} \mu_{jl}.$$

Here, $a^{(\boldsymbol{\tau})}$ captures cases 1-3 of the $K^{(p)}$ distance between $\boldsymbol{\tau}$ and input lists , while $b^{(\boldsymbol{\tau})}$ captures case 4 using the parameters defined herein. More specifically, if items i and j are simultaneously in $\boldsymbol{\tau}$, $b^{(\boldsymbol{\tau})}$ equals the number of input lists in which i and j are both absent, and if neither i nor j is present in $\boldsymbol{\tau}$, $b^{(\boldsymbol{\tau})}$ equals the number of input lists in which i and j are simultaneously present. Furthermore, the $K^{(p)}$ distance between two top-k lists is non-decreasing in p (Fagin *et al.*, 2003) and, therefore, the set of optimal objective values for all values of p forms a piecewise linear envelope. Proposition 10 demonstrates that the set of optimal objective values for all values of p forms a piecewise linear envelope; an example is depicted in Fig. 5.1. This result will be used to derive further insights of the effect of penalty parameter p on the solution of problem (5.3). Expressly, the problem has the property that if a solution is the consensus list for two distinct values of p, it will also be the consensus list for any value in between.

Corollary 3 *If $\boldsymbol{\tau}^*$ is the consensus top-k for two distinct penalty parameters p^1 and p^2 such that $0 \leq p^1 < p^2 \leq 1$, then $\boldsymbol{\tau}^*$ is also the consensus top-k for any $p^1 < p < p^2$.*

Corollary 3 is a direct outcome of Proposition 10. Corollary 3 can help overcome the difficulty of selecting the value of penalty p. For example, if $\boldsymbol{\tau}^*$ is the consensus top-k list for $p = 0$ and $p = 1$, then it is also the consensus top-k list for every value of p.

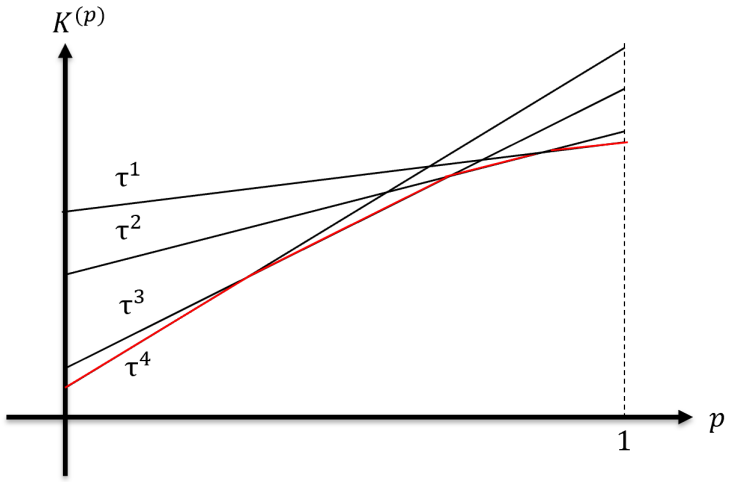

Figure 5.1: Example Depiction of Objective Values Obtained over All Values of p by Four Different Solution Top-k Lists (the Piecewise Linear Envelope Is Shown in Red).

One may also be interested in finding the consensus list with the minimum average $K^{(p)}$ distance to the input lists over all possible values of p for which the resulting function is a metric, i.e., $\forall p \in [0, 1]$, instead of only one specific value of p. We show that this robust solution is obtained by a specific value of p.

Proposition 11 *Given an instance of problem* (5.3)*, the consensus list obtained by using* $p = 0.5$ *has the least average distance* $K^{(p)}$ *to the input lists.*

We treat p as a random variable with a continuous uniform distribution over $[0, 1]$. The solution to the problem of finding a top-k list with the least average $K^{(p)}$ distance to the input lists with respect to all values of $p \in [0, 1]$ can be expressed as

$$\begin{aligned}
\boldsymbol{\tau}^* &= \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \ \mathbb{E}_{p \in [0,1]} \left[\sum_{l \in \boldsymbol{\mathcal{L}}} K^{(p)}(\boldsymbol{\tau}, \boldsymbol{\tau}^l) \right] \\
&= \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \ \mathbb{E}_{p \in [0,1]} \left[a^{(\boldsymbol{\tau})} + p b^{(\boldsymbol{\tau})} \right] \\
&= \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \ \left[a^{(\boldsymbol{\tau})} + \mathbb{E}_{p \in [0,1]}(p) b^{(\boldsymbol{\tau})} \right] \\
&= \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \ \left[a^{(\boldsymbol{\tau})} + 0.5 b^{(\boldsymbol{\tau})} \right]
\end{aligned}$$

$$= \arg\min_{\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}} \sum_{l \in \boldsymbol{\mathcal{L}}} K^{(0.5)}(\boldsymbol{\tau}, \boldsymbol{\tau}^l).$$

5.4 Polyhedral Comparison

Next, we compare the strength of the proposed MIPs based on their LP-relaxation models. First, we compare the strength of MIPs#2, #3, and #4. To that end, notice that these three MIPs become equivalent when $k \leq 2$—when the preference transitivity relations are irrelevant—or when $n = k$—when all items appear in the solution top-k list. For the remainder of the paper, let $\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3, \mathcal{P}^4$ be the polyhedral corresponding to the LP relaxations of MIPs#1, #2, #3, #4, respectively.

Theorem 11 *For any instance of* TOP-k-AGG, $\mathcal{P}^4 \subseteq \mathcal{P}^2 \subseteq \mathcal{P}^3$, *and these inclusions can be strict.*

Note that MIPs#2, #3, and #4 differ only in their preference transitivity constraints. First, we show that $\mathcal{P}^4 \subseteq \mathcal{P}^2 \subseteq \mathcal{P}^3$.

Since $0 \leq z_i \leq 1 \ \forall i \in \boldsymbol{\mathcal{I}}$, for every feasible solution in $\mathcal{P}^2, \mathcal{P}^3, \mathcal{P}^4$, we have that $(z_h + z_i + z_j)/3 \leq 1 \ \forall h, i, j \in \boldsymbol{\mathcal{I}},\ i, j > h,\ i \neq j$. Letting $(\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z})^{(4)} \in \mathcal{P}^4$ be a feasible solution to MIP#4, we have that

$$w_{hi}^{(4)} + w_{ij}^{(4)} + w_{jh}^{(4)} \leq 1 + \frac{1}{3}(z_i^{(4)} + z_j^{(4)} + z_h^{(4)}) \leq 2 \leq 3 - \frac{1}{3}(z_i^{(4)} + z_j^{(4)} + z_h^{(4)}).$$

Therefore, all feasible solutions to MIP#4 are also feasible to MIPs#2 and #3. Using the same logic, all feasible solutions to MIP#2 are feasible to MIP#3. This gives that $\mathcal{P}^4 \subseteq \mathcal{P}^2 \subseteq \mathcal{P}^3$.

To show that the inclusion $\mathcal{P}^4 \subseteq \mathcal{P}^2$ can be strict, consider a small instance with $\boldsymbol{\mathcal{I}} = \{1, 2, 3, 4\}$ and $k = 3$. Fix the solution $(\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z})^{(2)} \in \mathcal{P}^2$ as

$$x_{14}^{\prime(2)} = x_{24}^{\prime(2)} = x_{34}^{\prime(2)} = 0.24, \quad w_{12}^{(2)} = w_{23}^{(2)} = w_{31}^{(2)} = 0.62, \quad w_{14}^{(2)} = w_{24}^{(2)} = w_{34}^{(2)} = 0.38,$$

$$z_1^{(2)} = z_2^{(2)} = z_3^{(2)} = 0.81, \qquad z_4^{(2)} = 0.57;$$

with all other variables equal to 0. By inspection, this solution satisfies all constraints of MIP#2. However, we have that

$$w_{12}^{(2)} + w_{23}^{(2)} + w_{31}^{(2)} = 1.86 \nleq 1 + \frac{0.81 + 0.81 + 0.81}{3} = 1.81.$$

This indicates that this solution does not satisfy the preference transitivity constraints of MIP#4.

Next, we use a similar process to show that the inclusion $\mathcal{P}^2 \subseteq \mathcal{P}^3$ can be strict. Consider a small instance with $\boldsymbol{\mathcal{I}} = \{1, 2, 3, 4\}$ and $k = 3$. Fix the solution $(\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z})^{(3)} \in \mathcal{P}^3$ as

$$x_{14}'^{(3)} = x_{24}'^{(3)} = x_{34}'^{(3)} = 0.4, \qquad w_{12}^{(3)} = w_{23}^{(3)} = w_{31}^{(3)} = 0.7, \qquad w_{14}^{(3)} = w_{24}^{(3)} = w_{34}^{(3)} = 0.3,$$
$$z_1^{(2)} = z_2^{(3)} = z_3^{(3)} = 0.85, \qquad z_4^{(3)} = 0.45;$$

with all other variables equal to 0. By inspection, this solution satisfies all constraints of MIP#3. However, we have that

$$w_{12}^{(3)} + w_{23}^{(3)} + w_{31}^{(3)} = 2.1 \nleq 2.$$

This indicates that this solution does not satisfy the preference transitivity constraints of MIP#2. Next, we compare MIP#1 with MIPs#2, 3, 4. Since MIP#1 uses different variables than MIPs#2, 3, 4, their respective polytopes are in different dimensions and cannot be directly compared. However, since these formulations have a set of variables in common, $\boldsymbol{w}$, we compare the projections of their respective polytopes on the $\boldsymbol{w}$-space. For doing so, first we restate the definition of the projection operation.

Definition 18 *(Balas, 2005) Given a polyhedron of the form* $Q := \{(u, x) \in \mathbf{R}^p \times \mathbf{R}^q : Au + Bx \leq b\}$*, where* A, B *and* b *have* m *rows, the projection of* Q *onto* $\mathbf{R}^q$*, or onto*

the x-space, is defined as $proj_x Q := \{x \in \mathbf{R}^q : \exists u \in \mathbf{R}^p : (u, x) \in Q\}$.

Theorem 12 *For any instance of* TOP-k-AGG, $proj_{\boldsymbol{w}} \mathcal{P}^2, proj_{\boldsymbol{w}} \mathcal{P}^3, proj_{\boldsymbol{w}} \mathcal{P}^4 \subseteq proj_{\boldsymbol{w}} \mathcal{P}^1$, *and these inclusions can be strict.*

First, we prove that $\text{proj}_{\boldsymbol{w}}\mathcal{P}^3 \subseteq \text{proj}_{\boldsymbol{w}}\mathcal{P}^1$. We show that, starting from an arbitrary solution $(\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z}) \in \mathcal{P}^3$, we can deduce a solution $(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{w'}, \boldsymbol{w''}) \in \mathcal{P}^1$. To this end, we define the following affine mappings of variables from $\mathcal{P}^3$ to $\mathcal{P}^1$:

$$u_{it} = \frac{z_i}{k} \quad \forall i \in \boldsymbol{\mathcal{I}},\ \forall t \in \{1, \ldots, k\} \rightarrow \sum_{t=1}^{k} u_{it} = z_i \quad \forall i \in \boldsymbol{\mathcal{I}}, \tag{5.16a}$$

$$w'_{ij} = x'_{ij} \quad \forall i, j \in \boldsymbol{\mathcal{I}},\ i \neq j, \tag{5.16b}$$

$$w''_{ij} = x''_{ij} \quad \forall i, j \in \boldsymbol{\mathcal{I}},\ j > i. \tag{5.16c}$$

Mapping (5.16b)-(5.16c) guarantees that the objective function values achieved by the respective feasible points are equal. To establish that $\text{proj}_{\boldsymbol{w}}\mathcal{P}^3 \subseteq \text{proj}_{\boldsymbol{w}}\mathcal{P}^1$, it is sufficient to show that, given a feasible solution in $\mathcal{P}^3$, the mapped variables are guaranteed to satisfy all constraints of MIP#1 (i.e., this point belongs to $\mathcal{P}^1$).

Consider Constraint (5.8b). For any $t \in \{1, \ldots, k\}$, we have

$$\sum_{i \in \boldsymbol{\mathcal{I}}} u_{it} = \sum_{i \in \boldsymbol{\mathcal{I}}} \frac{z_i}{k} = \frac{\sum_{i \in \boldsymbol{\mathcal{I}}} z_i}{k} \xrightarrow{\sum_{i \in \boldsymbol{\mathcal{I}}} z_i = k} \sum_{i \in \boldsymbol{\mathcal{I}}} u_{it} = 1.$$

Therefore, mapping (5.16a) provides a solution that is guaranteed to satisfy Constraint (5.8b).

Consider Constraint (5.8c). For every $i \in \boldsymbol{\mathcal{I}}$, we have

$$\sum_{t=1}^{k} u_{it} = \sum_{t=1}^{k} \frac{z_i}{k} = \frac{k z_i}{k} = z_i \leq 1.$$

The last inequality follows from the fact that the $\boldsymbol{z}$-variables are binary. Therefore, mapping (5.16a) provides a solution that is guaranteed to satisfy Constraint (5.8c).

Next, consider Constraint (5.8d); we focus on the maximum value of the right-hand side of this constraint given mapping (5.16a). For any arbitrary item-pair (i, j)

and any $t \in \{1, \ldots, k-1\}$ we have

$$\begin{aligned}\sum_{t'=1}^{t} u_{it'} + \sum_{t''=t+1}^{k} u_{jt''} - 1 &= \sum_{t'=1}^{t} \frac{z_i}{k} + \sum_{t''=t+1}^{k} \frac{z_j}{k} - 1 \\ &= \frac{tz_i}{k} + \frac{(k-t)z_j}{k} - 1 \\ &\leq \frac{t}{k} + \frac{k-t}{k} - 1 = \frac{k}{k} - 1 = 1 - 1 = 0.\end{aligned}$$

The above equation states that using mapping (5.16a), the left-hand side values of Constraint (5.8d) will be non-positive. Since $w_{ij} \geq 0$, mapping (5.16a) provides a solution that is guaranteed to satisfy Constraint (5.8d).

Next, consider Constraint (5.8e). By summing over Constraint (5.10d), we have

$$2 \sum_{i,j \in \boldsymbol{\mathcal{I}}} w_{ij} \leq (k-1) \sum_{i \in \boldsymbol{\mathcal{I}}} z_i = k(k-1) \rightarrow \sum_{i,j \in \boldsymbol{\mathcal{I}}} w_{ij} \leq \frac{k(k-1)}{2},$$

which is exactly Constraint (5.8e).

Finally, consider Constraints (5.8f)-(5.8i). Mappings (5.16a)-(5.16c) imply that all feasible solutions to Constraints (5.10e)-(5.10h) are feasible to Constraints (5.8f)-(5.8i). Putting all pieces together, we have $\text{proj}_{\boldsymbol{w}} \mathcal{P}^3 \subseteq \text{proj}_{\boldsymbol{w}} \mathcal{P}^1$.

Note that the preference cycle-prevention constraints of MIP#3 have no counterpart in MIP#1. Therefore, we can show that the inclusion $\text{proj}_{\boldsymbol{w}} \mathcal{P}^3 \subseteq \text{proj}_{\boldsymbol{w}} \mathcal{P}^1$ can be strict by providing a solution that satisfies Constraints (5.10c)-(5.10f) but violates preference cycle-prevention Constraint (5.13), as this solution satisfies all constraints of MIP#1. There is an infinite number of such solutions; for example, consider a small instance with $\boldsymbol{\mathcal{I}} = \{1, 2, 3, 4\}$ and $k = 3$. Fix the solution $(\boldsymbol{w}, \boldsymbol{x'}, \boldsymbol{x''}, \boldsymbol{z})^{(3)}$ as

$$x_{14}'^{(3)} = x_{24}'^{(3)} = x_{34}'^{(3)} = 0.44, \quad w_{12}^{(3)} = w_{23}^{(3)} = w_{31}^{(3)} = 0.72, \quad w_{14}^{(3)} = w_{24}^{(3)} = w_{34}^{(3)} = 0.28,$$
$$z_1^{(2)} = z_2^{(3)} = z_3^{(3)} = 0.86, \qquad z_4^{(3)} = 0.42;$$

with all other variables equal to 0. By inspection, this solution satisfies Constraints

(5.10c)-(5.10f); however, it violates the preference transitivity constraints involved in MIP#3, as we have

$$w_{12} + w_{23} + w_{31} = 2.16 \not\leq 3 - (0.86 + 0.86 + 0.86)/3 = 2.14.$$

Finally, from Theorem 11, we have that $\mathcal{P}^4 \subseteq \mathcal{P}^2 \subseteq \mathcal{P}^3$; therefore, we can conclude that $\text{proj}_{\boldsymbol{w}} \mathcal{P}^2, \text{proj}_{\boldsymbol{w}} \mathcal{P}^4 \subseteq \text{proj}_{\boldsymbol{w}} \mathcal{P}^1$, and these inclusions can be strict.

5.5 Heuristic Methods

As the value of k and/or especially as the number of input lists increases, the universe of items can increase rapidly, which makes exact formulations computationally difficult or practically impossible to solve to optimality using exact methods, owing to the fact that the presented formulation has $O(n^3)$ constraints. Therefore, we adapt various heuristics defined for the rank aggregation problem, and we introduce a new greedy algorithm based on the proposed exact optimization models to find high-quality solutions. The rest of this section describes these methods.

BestInput: Similar to KEMENY-AGG (Ailon *et al.*, 2008), one can simply pick the input list with the least cumulative $K^{(p)}$ distance to the input lists. Stated otherwise, this method represents a simplified version of TOP-k-AGG where $\boldsymbol{\mathcal{T}} := \{\boldsymbol{\tau}^1, \ldots, \boldsymbol{\tau}^m\}$. It has a time complexity of $O(k^2 m^2)$.

Sort by Average Rank (SAR): This method sorts the items based on their average rank positions in the input lists, denoted by $\overline{\boldsymbol{\tau}}(i)$, and picks the k items with the lowest average ranks; ties are broken arbitrarily. Specifically, $\overline{\boldsymbol{\tau}}(i)$ is calculated as

$$\overline{\boldsymbol{\tau}}(i) = \frac{\sum_{l \in \boldsymbol{\mathcal{L}}} \boldsymbol{\tau}^l(i) \mathbb{1}_{i \in \boldsymbol{\tau}^l}}{\eta_i}. \tag{5.17}$$

Sort by Number of Appearance (SNA): This method sorts the items based on their number of appearances in the input lists, denoted by $\eta(i)$, and it picks the k

items with the highest number of appearances; ties are broken arbitrarily. Specifically, $\eta(i)$ is calculated as

$$\eta(i) = \sum_{l \in \boldsymbol{\mathcal{L}}} \mu_{il}. \tag{5.18}$$

Hybrid SAR-SNA (H-SA-SN): This hybrid method is the same as SAR with a modification that items with the same average rank are sorted by a non-increasing number of appearances.

Hybrid SNA-SAR (H-SN-SA): This hybrid method is the same as SNA with a modification that items with the same number of appearances are sorted by non-decreasing average ranks.

Heuristics SAR, SNA, H-SA-SN, and H-SN-SA all output a full list over $\boldsymbol{\mathcal{I}}$ whose top-k ranked items are returned as the solution. Their outputs are independent of the distance used to aggregate the top-k lists. Each of these four heuristics has a time complexity of $O\left(mk + n\right)$.

Spearman's Footrule: Fagin *et al.* (2003) proposed the only known solution method for TOP-k-AGG. It is a 2-approximation algorithm that uses the $F^{(k+1)}$ distance, in which all items that are not in a top-k list are tied together in position $k+1$. TOP-k-AGG using the $F^{(k+1)}$ distance can be solved in polynomial time via the minimum-cost perfect matching problem (Dwork *et al.*, 2001; Fagin *et al.*, 2003). This method outputs a full list over $\boldsymbol{\mathcal{I}}$ whose top-k items are returned as the solution. It has a time complexity of $O(mn^2)$.

Iterative Greedy Heuristic (IGH): This is a greedy iterative algorithm that uses objective function (5.10a) of the exact formulation featured herein; its pseudocode is presented in Algorithm 7 and summarized as follows. Placing item i at position t of the solution top-k list leads to an *imposed cost*, denoted by IC_i^t, on the objective function, where $1 \leq t \leq k$. The algorithm iteratively adds an item to the

top-k list, namely, the available item with the lowest IC_i^t value.

In greater detail, placing item i in first place of the solution top-k list imposes the cost

$$IC_i^1 = \sum_{j \in \boldsymbol{\mathcal{I}} \setminus \{i\}} s_{ji}.$$

The item with the lowest imposed cost is placed first in the solution top-k list; assume that item i_1 has the lowest imposed cost, and, hence, place i_1 at the first position (rank). In the second step, calculate the imposed cost of placing each item in the second position while holding item i_1 in first place, and place the item with the lowest imposed cost in the second position of the solution top-k list. More generally, assume that at step $1 \leq t \leq k$, items $i_1, i_2, \ldots, i_{t-1}$ have been placed in positions $1, 2, \ldots, t-1$, respectively. Placing item $i \in \boldsymbol{\mathcal{I}} \setminus \{i_1, i_2, \ldots, i_{t-1}\}$ in position t of the solution top-k list imposes the cost

$$IC_i^t = \sum_{j \in \boldsymbol{\mathcal{I}} \setminus \{i, i_1, i_2, \ldots, i_{t-1}\}} s_{ji} + p \sum_{l \in \boldsymbol{\mathcal{L}}} \sum_{r \in \{i, i_1, i_2, \ldots, i_{t-1}\}} (1 - \mu_{il})(1 - \mu_{rl}).$$

If there are multiple items with the same imposed cost at some iteration, one is selected arbitrarily. After the second step, the IC_i^t values can be easily updated in linear time for the remaining items. Hence, this method has a time complexity of $O(n^2)$.

A few last remarks on this heuristic are in order. Since it is not known which items will be in the top-k list until the conclusion of the algorithm, the third term in the objective function (5.10a) is dropped from computations (the contribution of this term in the objective function is relatively small compared to other terms, especially for small values of p). Additionally, the second term in the objective function is effectively neglected in the first iteration; however, it is added back in the subsequent iterations. As a post-processing subroutine for all of the presented heuristic

Algorithm 7: Iterative Greedy Heuristic (IGH)

Input : $p, \boldsymbol{\mathcal{I}}$, pairwise preference matrix, $[\mu_{il}] \in \mathbb{B}^{n \times |\boldsymbol{\mathcal{L}}|}$
Output: Solution top-k list ($\boldsymbol{\tau}$)

$\boldsymbol{\tau} \leftarrow \emptyset$;
for $t = 1, \ldots, k$ **do**
  $\bar{i} = \arg\min_{i \in \boldsymbol{\mathcal{I}} \setminus \boldsymbol{\tau}} IC_i^t = \arg\min_{i \in \boldsymbol{\mathcal{I}} \setminus \boldsymbol{\tau}} \sum_{j \in \boldsymbol{\mathcal{I}} \setminus (\boldsymbol{\tau} \cup \{i\})} p_{ji} + p \sum_{l \in \boldsymbol{\mathcal{L}}} \sum_{r \in \boldsymbol{\tau}} (1 - \mu_{il})(1 - \mu_{rl})\, \boldsymbol{\tau}$
  $\leftarrow \boldsymbol{\tau} \cup \bar{i}$;
  $\boldsymbol{\tau}(\bar{i}) = t$;
Return $\boldsymbol{\tau}$;

algorithms, we implement a variant of *local search* (Dwork *et al.*, 2001). This subroutine performs "single adjacent swap", where two adjacent items in the top-k list swap positions if doing so improves the objective function value. Letting $\boldsymbol{\tau}$ denote a solution top-k list output by a given heuristic, notice that changing the position of i and j, where $\boldsymbol{\tau}(j) = \boldsymbol{\tau}(i) + 1$, reduces the current objective function value only if $s_{ji} > s_{ij}$.

5.6 Data Reduction

Exact methods and even certain heuristics can become computationally demanding with increasing the size of the universe. Yet, it is straightforward to discern that, when k is small relative to n, a large number of items will have little relevance to the consensus top-k list. This observation is the motivation for the data reduction technique introduced in this section. The proposed technique discards items that are unlikely to appear in the consensus list in order to boost the computational viability of exact and certain inexact methods.

The proposed data reduction technique leverages parameters $\overline{\boldsymbol{\tau}}$ and η—the average rank and the number of appearances, respectively—to reduce the size of $\boldsymbol{\mathcal{I}}$. It identifies items that have a high $\overline{\boldsymbol{\tau}}$-value and a low η-value to be discarded, as these

items are unlikely to appear in the consensus top-k list. Let τ_{tr} and η_{tr} be the thresholds of $\overline{\boldsymbol{\tau}}$ and η-values of discarded items, respectively, and define $\boldsymbol{\mathcal{I}}_r$ as the subset of items identified by the data reduction to be discarded, that is,

$$\boldsymbol{\mathcal{I}}_r = \{i \in \boldsymbol{\mathcal{I}} : \overline{\boldsymbol{\tau}}(i) > \tau_{tr} \text{ and } \eta_i < \eta_{tr}\}.$$

As an example, consider the following top-4 lists:

$$\boldsymbol{\tau}^1 = \cdots = \boldsymbol{\tau}^6 = \{5, 6, 7, 8\},\ \boldsymbol{\tau}^7 = \{1, 2, 3, 4\}.$$

The consensus top-k list for $p < 1/3$ is $\{5, 6, 7, 8\}$, the consensus top-k list for $p > 2/3$ is $\{5, 6, 7, 1\}$, and both are consensus top-k lists for $p = 1/3$. Note that all items that appeared in the consensus list either have a high $\overline{\boldsymbol{\tau}}$-value, η-value, or both.

The size of $\boldsymbol{\mathcal{I}}_r$ is affected by the user-specified thresholds τ_{tr} and η_{tr}. Based on a set of independently conducted experiments, recommended settings are $\tau_{tr} = k/3$ and $\eta_{tr} = m/3$; however, these values can be changed depending on the size of $\boldsymbol{\mathcal{I}}_r$ and the context at hand. It is important to mention that the outputs of the data reduction technique may deviate from the optimal solution to Problem (5.3), meaning that the items in $\boldsymbol{\mathcal{I}}_r$ might appear in the consensus top-k list. Nonetheless, discarding these items typically leads to no more than minor differences with the outputs of exact methods, while it can result in significant improvements in the run time of the proposed techniques, especially for large-scale problem instances.

We propose two different ways to handle items identified by the proposed data reduction technique: *Explicit reduction* and *implicit reduction*. In explicit reduction, all items are completely removed from the universe of items, resulting in a new smaller problem. This removal could impact other items in $\boldsymbol{\mathcal{I}} \backslash \boldsymbol{\mathcal{I}}_r$ that could end up in the consensus top-k list. In implicit reduction, all items in $\boldsymbol{\mathcal{I}}_r$ are restricted from appearing in the solution top-k list, but they factor into the calculations of other

eligible items that may appear.

Implicit reduction may be implemented on certain exact and inexact solution approaches as follows. In the exact formulation, the constraint $z_i = 0$ is added $\forall i \in \boldsymbol{\mathcal{I}}_r$; as a result, all preference cycle prevention constraints that involve any discarded items become redundant and are omitted from the formulation. The implicit (and explicit) reduction has no impact on BestInput; moreover, only explicit reduction can be applied to Spearman's footrule. Finally, the implicit reduction is implemented in IGH by excluding items $\boldsymbol{\mathcal{I}}_r$ as candidates from the solution top-k list, i.e., by modifying line 3 of Algorithm 7.

5.7 Computational Experiments

This section compares the solution quality and run time of the different solution methods presented in Sections 5.3-5.6. More specifically, we compare the run time and solution quality of MIP#1, MIP#4, MIP#4 with implicit reduction (MIP#4_Implicit), MIP#4 with explicit reduction (MIP#4_Explicit), BestInput, Spearman's footrule (Spearman), SAR, SAN, H-SA-SN, H-SN-SA, IGH, IGH with implicit reduction (IGH_Implicit), and IGH with explicit reduction (IGH_Explicit). All experiments were carried out on a computer with an Intel(R) Core(TM) i7-7700 CPU 3.60GHz with 32 GB RAM. The optimization models were solved using CPLEX solver version 12.10.0 to a relative optimality gap of 1% or until a 600-second time limit is reached. The rest of this section is organized as follows. Section 5.7.1 describes the data set, and Section 5.7.2 presents the computational results.

5.7.1 Data Set

We use Mallow's model proposed in Chierichetti *et al.* (2018) to generate synthetic top-k lists (see Section 5.2.1). Chierichetti *et al.* (2018) proposed two sampling

methods for generating input lists from this model: an exact sampling method using dynamic programming with time complexity of $O\left(k^2 4^k + k^2 \log n\right)$, and an approximate sampling method using a Markov chain model with a relaxed time bound of $O\left(k^5 \log k\right)$. To describe the steps of the Markov chain method used herein, let $\boldsymbol{\tau} \in \boldsymbol{\mathcal{T}}$ be a randomly selected top-k list, τ_t^{-1} denote the item in position t of $\boldsymbol{\tau}$, $\boldsymbol{\tau}^c = \boldsymbol{\mathcal{I}} \backslash \boldsymbol{\tau}$ be the set of items not in $\boldsymbol{\tau}$, and $\boldsymbol{\tau}'$ be the ground-truth list. Two items i and j are said to be $\boldsymbol{\tau}'$-adjacent if there is no item h such that $\boldsymbol{\tau}'(i) < \boldsymbol{\tau}'(h) < \boldsymbol{\tau}'(j)$ or $\boldsymbol{\tau}'(j) < \boldsymbol{\tau}'(h) < \boldsymbol{\tau}'(i)$. Let $\mathcal{C}$ be a Markov chain on a state space $\boldsymbol{\Omega} = \boldsymbol{\mathcal{T}}$ (the states of this Markov Chain are the possible top-k lists). Additionally, let $\mathcal{C}(s_1, s_2)$ denote the transition probabilities from state $s_1 \in \mathcal{C}$ to state $s_2 \in \mathcal{C}$; $\mathcal{C}$ is an ergodic chain, guaranteeing convergence to the unique stationary distribution (Chierichetti *et al.*, 2018). The Markov chain $\mathcal{C}$ is defined as follows. Choose $1 \leq t \leq k-1$ u.a.r. and equiprobably do one of (Chierichetti *et al.*, 2018):

1. Transposition step: Equiprobably do one of:

   (a) If $\tau_t^{-1} \in \boldsymbol{\tau}'$, find minimum $v > t$ such that $\tau_v^{-1} \in \boldsymbol{\tau}'$, and put them in the order of $\boldsymbol{\tau}'$ w.p. $e^\beta/(1+e^\beta)$ and the opposite order w.p. $1/(1+e^\beta)$.

   (b) If $\tau_t^{-1} \notin \boldsymbol{\tau}'$, find minimum $v > t$ such that $\tau_v^{-1} \notin \boldsymbol{\tau}'$ and put them in the order of $\boldsymbol{\tau}$ w.p. 0.5 and the opposite order w.p. 0.5.

   (c) If ($\tau_t^{-1} \in \boldsymbol{\tau}'$ and $\tau_{t+1}^{-1} \notin \boldsymbol{\tau}'$) or ($\tau_t^{-1} \notin \boldsymbol{\tau}'$ and $\tau_{t+1}^{-1} \in \boldsymbol{\tau}'$), put them in the order of $\boldsymbol{\tau}'$ w.p. $e^\beta/(1+e^\beta)$ and in the opposite order w.p. $e^\beta/(1+e^\beta)$.

2. Substitution step: W.p. 0.5 stay at the current state, and w.p. 0.5 equiprobably do one of:

   (a) A homogeneous substitution:

i. If $\tau_t^{-1} \in \boldsymbol{\tau}'$, let j be the $\boldsymbol{\tau}'$-adjacent item to τ_t^{-1} such that $\boldsymbol{\tau}'(j) > \boldsymbol{\tau}'(\tau_t^{-1})$ and if $j \notin \boldsymbol{\tau}$, replace τ_t^{-1} by j w.p. $e^{\beta}/(1+e^{\beta})$. If $\tau_t^{-1} \in \boldsymbol{\tau}'$, let j be the $\boldsymbol{\tau}'$-adjacent item to τ_t^{-1} such that $\boldsymbol{\tau}'(j) < \boldsymbol{\tau}'(\tau_t^{-1})$ and if $j \notin \boldsymbol{\tau}$, replace τ_t^{-1} by j w.p. $1/(1+e^{\beta})$.

ii. If $\tau_t^{-1} \notin \boldsymbol{\tau}'$, pick item c u.a.r. from $\boldsymbol{\tau}^c$, and if $c \notin \boldsymbol{\tau}'$, replace τ_t^{-1} by item c w.p. 0.5.

(b) A non-homogeneous substitution: choose item c u.a.r. from $\boldsymbol{\tau}^c$ and compare it with τ_k, if only one of them is in $\boldsymbol{\tau}'$, keep the $\boldsymbol{\tau}'$ item inside w.p. $e^{\beta(1+p.i)}/(1+e^{\beta(1+p.i)})$ and the item outside $\boldsymbol{\tau}'$ w.p. $1/(1+e^{\beta(1+p.i)})$ where $i = |\tau[1,k-1] \cap \boldsymbol{\tau}'^c|$.

If the premise is not satisfied in any of the above, do nothing.

Similar to Chierichetti *et al.* (2018), we use $\beta = 1$; note that using small values of β translates to introducing more noise in the generated data. We use 1,000 iterations of the Markov chain, and we set the ground truth as the identity top-k list, i.e., $\boldsymbol{\tau}' = \{1, 2, \ldots, k\}$. Additionally, let $m = |\mathcal{L}|$ denote the number of input lists. We use different settings of three parameters, k, m, p to generate instances; for each combination of k and m values, we use $p \in \{0, 0.5, 1\}$. The initial size of the universe is set to mk for the purpose of sampling to allow the lists to be less cohesive collectively, which makes aggregation a much more difficult task; however, the precise size is typically much smaller and is determined after sampling, as some items do not appear in any of the generated lists. Note that the $k \leq n \leq mk$; we have $n = k$ whenever exactly k items appear in all list, and $n = mk$ whenever all items in all lists are different. The thresholds of the data reduction technique were initially set to $\tau_{tr} = k/3$ and $\eta_{tr} = m/3$; however, if the size of remaining items was less than k, these values were increased to $\tau_{tr} = k/5$ and $\eta_{tr} = m/5$. All reported numbers are

rounded to two decimal points.

5.7.2 Computational Results

The computational results compare the featured methods based on the cumulative $K^{(p)}$ distance of the aggregated list to all input lists (i.e., their objective function values). The experiments are divided into three parts. The first part compares MIPs#1 and #4. The second part compares MIP#4 and the introduced heuristics on small- to medium-sized instances. After evaluating the results, the third part selects the best-performing methods from the second part for further testing on larger instances. For comparing the solution quality of different methods, we define the %Relative_Gap of a method x as the difference between objective function values of method x from the best found objective function value, divided by best found objective function value.

For the first experiment, the tested generating parameters are $(k = 10, m = 5, p)$, $(k = 10, m = 20, p)$, and $(k = 25, m = 5, p)$, where $p \in \{0, 0.5, 1\}$. For each of the nine nine resulting parameter configurations, 20 instances are generated. Table 5.1 reports the average size of the universe and the geometric mean run time of MIPs#1 & #4.

Overall, MIP#4 had a dominant performance: It either achieved the optimal solutions faster or achieved a higher-quality solution by the time of termination. The biggest differences are highlighted by cases $(k = 10, m = 5, p = 1)$, where MIP#4 was more than 60x faster than MIP#1, and $(k = 25, m = 5, p = 1)$, where MIP#4 had a geometric run time of 173.34 and MIP#1 had a %Relative_Gap of 147.59 after 623.33 seconds.

For the second experiment, the tested generating parameters are $(k = 10, m = 20, p)$, $(k = 25, m = 5, p)$, and $(k = 50, m = 5, p)$, where $p \in \{0, 0.5, 1\}$. For each of

Table 5.1: Geometric Mean of Run Time (in Seconds) And %Relative_Gap of MIPs#1 and #4

	n	MIP#1		MIP#4	
		Time	%Relative_Gap	Time	%Relative_Gap
$k = 10, m = 5, p = 0$	25.65	601.65	1.41	1.63	0 (118.55)
$k = 10, m = 5, p = 0.5$	23.4	207.67	0 (95.8)	1.72	0 (95.8)
$k = 10, m = 5, p = 1$	23.25	80.06	0 (91.05)	1.34	0 (91.05)
$k = 10, m = 20, p = 0$	112.05	659.07	1.15	619.37	0 (1,226)
$k = 10, m = 20, p = 0.5$	115.30	610.71	0.62	415.85	0 (949.35)
$k = 10, m = 20, p = 1$	116.40	615.51	1.82	373.58	0 (1,079.3)
$k = 25, m = 5, p = 0$	81.75	631.84	31.09	620.74	0 (1,564.05)
$k = 25, m = 5, p = 0.5$	80.9	623.89	122.18	214.31	0 (1455.75)
$k = 25, m = 5, p = 1$	85.55	623.33	147.59	173.34	0 (1661.7)

the nine nine resulting parameter configurations, 20 instances are generated. Table 5.2 reports the average size of the universe and the average size of $|\boldsymbol{\mathcal{I}}_r|$ (average number of discarded items). It also reports the average objective function value achieved by each tested method. The latter are reported in terms of the percentage difference to the best performing method (i.e., the relative gap); in other words, a value of zero under these columns indicates the best performing method (with the corresponding objective value shown in parentheses) along that specific each row (i.e., parameter setting). Table 5.3 reports the geometric mean run times of tested methods. Highlights of the obtained results are as follows. MIP#4 achieved the lowest %Relative_Gap for four data sets, IGH for four data sets, IGH_Implicit for three data sets, and Exact_Implicit for one data set. H-SN-SA dominated other sorting-based methods (SAR, SNA, H-SA-SN) in terms of solution quality while maintaining a similar run time. Whenever MIP#4 did not terminate within the time limit, the best solution found up to that point was used in the computations.

Data reduction was helpful for exact methods; it reduced the run time or improved the solution quality or both. The effect of data reduction on IGH was highly dependent on the data set; IGH_Implicit outperformed IGH_Explicit in terms of so-

lution quality, whereas the reverse was true for run time. To round out the results, Spearman's footrule and BestInput produced neither high-quality solutions nor low run times. Based on the discussed results, we selected IGH, IGH_Implicit,

Table 5.2: %Relative_Gap of The Tested Methods

	n	$\lvert\mathcal{I}_r\rvert$	BestInput	Spearman	SAR	SNA	H-SA-SN	H-SN-SA	IGH	IGH_Implicit	IGH_Explicit	MIP#4	MIP#4_Implicit	MIP#4_Explicit
$k = 50, m = 5, p = 0$	165.6	70.4	20.65	35.53	25.56	10.45	25.47	7.52	0 (6301.65)	0 (6301.65)	20.93	25.03	15.18	16.52
$k = 50, m = 5, p = 0.5$	162.35	69.15	30.78	5.37	39.38	6.91	39.02	4.82	0 (7829.8)	0 (7829.8)	11.66	26.37	15.38	16.67
$k = 50, m = 5, p = 1$	161.3	68.65	38.93	46.48	48.67	5.45	48.60	3.76	0 (9329.6)	0 (9329.6)	13.13	27.86	13.87	15.72
$k = 10, m = 20, p = 0$	99.0	86.6	16.96	38.49	25.47	0.76	24.93	0.45	0 (972.4)	0.36	2.81	0.28	2.66	2.75
$k = 10, m = 20, p = 0.5$	89.9	61.7	28.03	45.16	31.02	1.51	31.07	0.92	0.30	0.63	1.65	0 (689.3)	1.13	1.02
$k = 10, m = 20, p = 1$	65.95	55.75	29.34	45.96	0.45	0.89	1.87	35.88	1.86	35.84	1.17	0 (704.3)	0.99	0.99
$k = 25, m = 5, p = 0$	75.25	36.65	26.80	39.42	28.49	12.40	28.32	8.71	1.33	1.33	12.78	6.94	0 (1243.7)	0.38
$k = 25, m = 5, p = 0.5$	66.25	29.75	36.97	48.08	51.83	51.38	51.75	5.05	1.49	1.49	5.20	0 (1306.05)	0.18	0.46
$k = 25, m = 5, p = 1$	67.7	31.15	42.56	47.99	61.35	7.19	61.28	5.00	1.09	1.09	5.50	0 (1564.9)	0.17	0.54

Table 5.3: Geometric Mean of Run Time of the Tested Methods (in Seconds)

	BestInput	Spearman	SAR	SNA	H-SA-SN	H-SN-SA	IGH	IGH_Implicit	IGH_Explicit	MIP#4	MIP#4_Implicit	MIP#4_Explicit
$k = 50, m = 5, p = 0$	0.17	0.90	0.00	0.00	0.00	0.00	0.03	0.03	0.03	736.56	736.56	634.92
$k = 50, m = 5, p = 0.5$	0.19	0.97	0.00	0.00	0.00	0.00	0.03	0.02	0.02	747.58	714.66	633.93
$k = 50, m = 5, p = 1$	0.19	0.95	0.00	0.00	0.00	0.00	0.04	0.03	0.03	784.54	743.44	635.63
$k = 10, m = 20, p = 0$	0.07	0.60	0.00	0.00	0.00	0.00	0.02	0.01	0.01	633.68	37.87	6.65
$k = 10, m = 20, p = 0.5$	0.05	0.31	0.00	0.00	0.00	0.00	0.01	0.01	0.01	83.29	12.98	0.3
$k = 10, m = 20, p = 1$	0.04	0.26	0.00	0.00	0.00	0.00	0.01	0.01	0.01	45.95	9.61	0.15
$k = 25, m = 5, p = 0$	0.02	0.10	0.00	0.00	0.00	0.00	0.01	0.01	0.01	612.48	107.33	317.69
$k = 25, m = 5, p = 0.5$	0.02	0.11	0.00	0.00	0.00	0.00	0.01	0.01	0.01	154.67	25.98	36.81
$k = 25, m = 5, p = 1$	0.02	0.10	0.00	0.00	0.00	0.00	0.01	0.01	0.01	40.55	22.07	22.38

IGH_Explicit, and H-SN-SA for further testing on larger instances in the third part of the experiments. Specifically, we tested these algorithms on generated instances with the following characteristics: $(k = 125, m = 20, p)$, $(k = 500, m = 5, p)$, and $(k = 1000, m = 20, p)$, where $p \in \{0, 0.5, 1\}$. For each of the nine nine resulting parameter configurations, 20 instances are generated. Tables 5.4 and 5.5 report the same statistics as Tables 5.2 and 5.3, respectively. In the second part, IGH achieved the best performance in all data sets; it shared the best performance with IGH_Implicit in four data sets. H-SN-SA had a very good performance, differing by less than 1% from IGH in four data sets. However, IGH and its variant had a geometric mean of 3.43 seconds or higher, whereas H-SN-SA took 0.00 seconds. To round out the results, IGH_Explicit was dominated by H-SN-SA, both in terms of solution quality and run time. Most impressively, IGH had the best performance similar to the second experiment. As a final note, the local search subroutine was able to improve the

Table 5.4: %Relative_Gap of The Tested Methods

	n	$\lvert\mathcal{I}_r\rvert$	H-SN-SA	IGH	IGH_Implicit	IGH_Explicit
$k = 125, m = 20, p = 0$	1,976	1,381.55	0.86	**0** (219,729.45)	1.23	4.24
$k = 125, m = 20, p = 0.5$	1,935	1,395.65	0.43	**0** (317,769.9)	2.39	3.31
$k = 125, m = 20, p = 1$	2,011	1,421.95	0.40	**0** (409,687)	3.31	4.57
$k = 500, m = 5, p = 0$	2,043	1,092.6	1.76	**0** (762,631.9)	**0** (762,631.9)	12.16
$k = 500, m = 5, p = 0.5$	2,062	1,095.9	4.89	**0** (1,000,518)	0.00	11.09
$k = 500, m = 5, p = 1$	2,086.5	1,098.6	1.14	**0** (1,238,945.6)	0.00	13.35
$k = 1000, m = 5, p = 0$	4,170	2,177.4	1.64	**0** (3,054,883.2)	**0** (3054883.2)	11.81
$k = 1000, m = 5, p = 0.5$	4,096	2,202.1	0.71	**0** (4,026,861.8)	**0** (4026861.8)	11.08
$k = 1000, m = 5, p = 1$	4,024	2,190.75	1.12	**0** (4,975,294.0)	0 (4975294.0)	13.38

Table 5.5: Geometric Mean of Run Time of the Tested Methods (in Seconds)

	H-SN-SA	IGH	IGH_Implicit	IGH_Explicit
$k = 125, m = 20, p = 0$	0.00	3.47	3.45	3.43
$k = 125, m = 20, p = 0.5$	0.00	3.45	3.43	3.40
$k = 125, m = 20, p = 1$	0.00	3.58	3.57	3.54
$k = 500, m = 5, p = 0$	0.00	3.94	3.90	3.81
$k = 500, m = 5, p = 0.5$	0.00	3.69	3.66	3.57
$k = 500, m = 5, p = 1$	0.00	3.92	3.87	3.80
$k = 1000, m = 5, p = 0$	0.00	15.65	15.19	14.59
$k = 1000, m = 5, p = 0.5$	0.00	15.32	15.02	14.55
$k = 1000, m = 5, p = 1$	0.00	15.56	15.22	14.76

solution quality of the aggregated list of the heuristic algorithms by up to 1.82%. This subroutine had the most effect on SAR and SAN methods; however, its improvement on all other methods was less than 1%. Nevertheless, the required time for applying this subroutine was completely negligible.

All things considered, IGH had the best performance in terms of solution quality; however, its run time rapidly increased with the size of the problem. On the other hand, H-SN-SA yielded competitive solutions (differing by less than 8% from IGH), but in far less time. Most impressively, H-SN-SA is a distance-free aggregation technique, and it is robust to the size of the problem. This feature makes it very suitable for solving very large instances.

5.8 Case Study: University Rankings

To assess the practical implications of TOP-k-AGG, we study the university ranking problem. Many university rankings have been proposed in recent years including the U.S. News and World Report Best Global Universities (USNEWS) (USN, 2022), the Times Higher Education World University Rankings (TIMES) (TIM, 2022), the Quacquarelli Symonds World University Rankings (QS) (QS, 2022), and the Academic Ranking of World Universities (ARWU) (ARW, 2022a). University rankings have many implications for universities and their students. They impact prospective students, especially international students, who use them to identify the universities where they will submit an application (Bowman and Bastedo, 2009; Çakır *et al.*, 2015). Additionally, they have implications on the status of higher education institutions and their external funding, as universities use rankings to promote their educational excellence. Naturally, a higher rank attracts more funding and high-quality scholars. However, there is a discrepancy between the criteria and associated weights used by each ranking institution (Çakır *et al.*, 2015). U.S. News utilizes 13 criteria: research reputation (12.5%), regional research reputation (12.5%), publications (10%), books (2.5%), conferences (2.5%), normalized citation impact (10%), total citations (7.5%), number of publications that are among the 10% most cited (12.5%), percentage of total publications that are among the 10% most cited (10%), international collaboration – relative to country (5%), international collaboration (5%), number of highly cited papers that are among the top 1% most cited in their respective field (5%), percentage of total publications that are among the top 1% most highly cited papers (5%) (Morse and Castonguay, 2021). TIMES utilizes 5 main criteria: teaching (reputation survey, staff-to-student ratio, doctorate-to-bachelor's ratio, doctorates-awarded-to-academic-staff ratio, institutional income) (30%), research

(reputation survey, research income, research productivity) (30%), citations (30%), international outlook (proportion of international students, proportion of international staff, international collaboration) (7.5%), and industry income (2.5%) (Ross, 2021). QS utilizes 5 main criteria: academic reputation (40%), faculty/student ratio (20%), employer reputation (10%), citations per faculty (10%), international student ratio & international faculty ratio (10%) (Laura, 2022). Finally, ARWU utilizes 5 main criteria: quality of education (10%), quality of faculty (20%), research output (40%), and per capita performance (10%) (ARW, 2022b).

There is an ongoing debate on variable selection, weights, and variable interdependencies involved in the difference university ranking methodologies(Çakır *et al.*, 2015). Some recent studies have addressed the university ranking aggregation problem using score-based methods. Tavanaei *et al.* (2018) addressed this problem by minimizing the cumulative deviation of the rank of each university in the aggregate list from its average rank in the input lists. Furthermore, Zhang *et al.* (2021) proposed sorting universities based on their out-in degree ratio in the *university graph*, where each node represents a university, and there is a directed arc from node i to node j if university i has been ranked ahead of university j in more than half of the input lists.

The goal of this study is to identify the top-100 U.S. universities by using Top-k-Agg to aggregate the USNEWS, QS, TIMES, and ARWU obtained from their websites. The input lists are truncated to the first 100 universities, and the universe of items, $\boldsymbol{\mathcal{I}}$, is formed by considering all universities that have appeared at least once in the top-100 positions of the input lists. Furthermore, if two universities i and j are tied in an input list, this is assumed to contribute 0.5 to s_{ij} and s_{ji} instead of 1. The rationale for doing this is as follows. We can break the ties in the input lists by considering all strict rankings involving any permutation of the tied items and

redistributing the unit weight of each input list over the created lists. In doing so, if i and j are tied in one input list, i will be ranked ahead of j in half of the created strict rankings and j in the other half. For example, consider a top-4 list $\boldsymbol{\tau} = \{a \approx b, c \approx d\}$ (a and b are tied but ranked ahead of c and d who are also tied) with a unit weight. We can break the ties by creating four strict lists: $\boldsymbol{\tau}^1 = \{a, b, c, d\}$, $\boldsymbol{\tau}^2 = \{a, b, d, c\}$, $\boldsymbol{\tau}^3 = \{b, a, c, d\}$, $\boldsymbol{\tau}^4 = \{b, a, d, c\}$, each with a weight of 0.25. Note that a is ranked ahead of b in half of the created lists, and b is ranked of a in the other half; the same is true for c and d. A similar approach is taken for dealing with items tied at the bottom of the list while truncating the lists. For example, consider the 2022 ARWU list where 85 universities occupy the first 85 positions and 21 universities are tied for positions 85-106, clearly, only 15 of the 21 tied universities can be included in the top-100 list. To describe how such lists are truncated, let n' be the number of tied universities at the bottom of top-100 list where only $k' < n'$ of them can be included in the top-100 list. Similar to the above treatment of ties, we can consider all strict rankings involving any permutation of k' out of n' items at the bottom of the list by redistributing the unit weight of each input list over the created lists, that is, create $\binom{n'}{k'}k'!$ strict lists, each with a weight of $1/\binom{n'}{k'}k'!$. Two universities i and j both will be both present and i will be ranked ahead of j in $0.5\binom{n'-2}{k'-2}k'!$ of the created lists; i, but not j, will be present in $\binom{n'-2}{k'-1}k'!$ of the created lists; and neither i nor j will be present in $\binom{n'-2}{k'}k'!$ of the created lists. This will allow us to process the data without actually creating all the possible strict lists, which can be in the order of billions. Furthermore, the penalty parameter p is set to 0.5, mainly due to Proposition 11. After obtaining the consensus list, a subset of adjacent universities is tied if any permutation of them, while keeping the rest of the orderings the same, does not change the optimal objective function value. The complete input lists and the consensus top-100 U.S. universities in 2022, 2021, and 2020 are displayed in Tables

5.6-5.11

Perhaps, the most interesting parts of the results are that the Harvard University, Stanford University, and MIT are tied as the highest-ranked universities in the U.S. in 2020; Harvard University and Stanford University are tied as the highest-ranked universities in the U.S. in 2021; but only Harvard University is identified as the highest-ranked university in the U.S. in 2022.

As a final note, the $K^{(p)}$ distance, # of full rank reversals, and # of partial rank reversals between the four input lists, and between each input list and the consensus list for 2022, 2021, and 2020 are reported in Tables 5.12-5.14. Interestingly, for all three years, USNEWS and ARWU are similar to each other, as are TIMES and QS (they have the lowest $K^{(p)}$ distance, # of full rank reversals, and # of partial rank reversals to each other compared to other lists). Furthermore, for all three years, USNEWS is the most similar input list to the consensus list, i.e., it is the solution to the BestInput method (see Section 5.5)).

Table 5.6: Global Rank of U.S. Universities That Have Appeared at Least Once in the Input 2022 Top-100 U.S. Universities Lists

University	US News	Times	QS	ARWU
Harvard University	1	2	5	1
Massachusetts Institute of Technology	2	5	1	3
Stanford University	3	4	3	2
University of California–Berkeley	4	8	32	5
Columbia University	6	11	19	8
University of Washington	7	29	85	17
California Institute of Technology	9	2	6	9
Johns Hopkins University	9	13	25	14
University of California–San Francisco	11	−−	−−	19
Yale University	12	9	14	11
University of Pennsylvania	13	13	13	15
University of California–Los Angeles	14	20	40	13
University of Chicago	15	10	10	10
Princeton University	16	7	20	6
University of Michigan–Ann Arbor	19	24	23	28
University of California–San Diego	21	34	48	21
Cornell University	22	22	21	12
Duke University	23	23	52	31
Northwestern University	24	24	30	30
New York University	30	26	42	25
Washington University in St. Louis	31	51	107	27
University of North Carolina–Chapel Hill	39	52	100	29
University of Pittsburgh	42	140	163	82
University of Texas–Austin	43	47	67	37
Ohio State University–Columbus	52	85	120	101
University of Wisconsin–Madison	52	58	75	33
University of Minnesota–Twin Cities	55	86	186	44
Icahn School of Medicine at Mount Sinai	57	−−	−−	101
Georgia Institute of Technology	58	45	88	151
University of Maryland–College Park	60	93	158	50
University of Colorado–Boulder	62	158	251	51
Boston University	65	62	112	101
University of California–Davis	67	67	108	101
University of California–Santa Barbara	67	68	146	57
University of Southern California	70	63	112	53
University of Illinois–Urbana-Champaign	72	48	82	49
Vanderbilt University	73	113	218	64

University	US News	Times	QS	ARWU
Emory University	74	82	160	101
Pennsylvania State University–University Park	80	119	96	101
University of California–Irvine	86	98	232	61
Rockefeller University	89	−−	−−	44
University of Arizona	99	150	268	101
University of Florida	99	154	173	94
Carnegie Mellon University	102	28	53	101
University of California–Santa Cruz	103	201	347	151
Michigan State University	108	93	157	151
University of Virginia	110	127	226	201
Brown University	119	64	60	99
University of Texas Southwestern Medical Center–Dallas	124	−−	−−	52
Purdue University–West Lafayette	127	105	116	83
Rutgers, The State University of New Jersey–New Brunswick	130	190	264	101
Baylor College of Medicine	135	−−	−−	151
Texas A&M University–College Station	140	193	168	151
Indiana University–Bloomington	141	167	311	101
Case Western Reserve University	144	126	161	101
University of Alabama–Birmingham	147	−−	−−	301
University of Massachusetts–Amherst	148	201	246	151
Oregon Health and Science University	151	251		201
University of Utah	151	251	358	101
University of Rochester	161	142	154	151
Arizona State University–Tempe	165	132	216	101
Rice University	167	136	94	101
University of California–Riverside	169	251	403	201
University of Iowa	174	251	455	201
Northeastern University	176	168	340	201
University of Colorado Anschutz Medical Campus	176	251	−−	−−
University of Cincinnati	191	−−	601	301
Stony Brook University–SUNY	195	301	378	301
Tufts University	198	172	275	151
University of Tennessee	218	301	541	201
Florida State University	223	251	475	201
North Carolina State University–Raleigh	227	301	300	201
University of Illinois–Chicago	227	301	285	301

University	US News	Times	QS	ARWU
Iowa State University	237	401	494	501
University of Miami	241	201	311	201
University of Oregon	244	351	651	301
Dartmouth College	247	99	191	301
George Washington University	247	201	355	201
Virginia Tech	250	251	346	201
Yeshiva University	250	−−	252	−−
Colorado State University	266	351	431	201
University of Maryland–Baltimore	266	−−	−−	301
University of Texas Health Science Center–Houston	266	−−	−−	301
University at Buffalo–SUNY	280	251	388	301
University of Kansas	280	401	387	201
University of Notre Dame	284	183	222	301
University of New Mexico	289	251	651	501
Oregon State University	295	−−	531	201
University of South Florida	298	201	581	201
Washington State University	302	351	427	301
Temple University	303	301	751	401
University of Georgia	303	351	541	301
University of Texas–Dallas	308	351	477	301
Georgetown University	314	130	248	401
Brandeis University	320	251	455	301
University of Connecticut	324	401	373	301
University of South Carolina–Columbia	330	401	601	401
University at Albany–SUNY	339	351	751	601
University of Delaware	346	301	531	201
University of Kentucky	348	501	651	301
Drexel University	354	351	651	301
University of Colorado–Denver	360	414	−−	201
Wayne State University	361	401	511	501
University of Nebraska–Lincoln	367	401	571	201
Baylor University	376	801	1001	501
University of Hawaii–Manoa	376	251	340	301
University of Massachusetts–Worcester	379	201	−−	−−
Syracuse University	383	351	651	701
Tulane University	417	401	436	601
University of Massachusetts–Boston	417	201	651	901
Virginia Commonwealth University	417	−−	701	201
University of Missouri	425	501	476	201

University	US News	Times	QS	ARWU
Wake Forest University	425	301	429	501
University of Houston	439	601	701	201
George Mason University	453	251	801	201
Rush University	461	301	−−	501
William & Mary	541	301	651	−−
Colorado School of Mines	577	401	461	701
Boston College	625	251	494	401
Rensselaer Polytechnic Institute	647	501	431	401
University of California–Merced	718	301	401	
Illinois Institute of Technology	729	351	444	901
Howard University	736	251	651	−−
Missouri University of Science and Technology	821	401	511	901
Nova Southeastern University	870	301	−−	−−
Lehigh University	890	601	531	801
University of Massachusetts–Dartmouth	1035	201	−−	−−
University of Massachusetts–Lowell	1141	201	−−	901
Hofstra University	1237	301	−−	−−
Clark University	−−	501	601	−−
The New School	−−	801	561	−−
Smith College	−−	−−	601	−−
The University of Texas M. D. Anderson Cancer Center	−−	−−	−−	71
Mayo Clinic Alix School of Medicine	−−	−−	−−	101
University of Massachusetts Medical School - Worcester	−−	−−	−−	201

Table 5.7: 2022 Consensus Top-100 U.S. Universities

University	Rank
Harvard University	1
Massachusetts Institute of Technology	2
Stanford University	3
University of California–Berkeley	4
California Institute of Technology	4
Columbia University	6
University of Chicago	6
Yale University	6
Princeton University	6
Johns Hopkins University	10
University of Pennsylvania	11
University of California–Los Angeles	12
Cornell University	12
University of Michigan–Ann Arbor	14
University of Washington	14
Northwestern University	16
University of California–San Diego	16
New York University	18
Duke University	18
Washington University in St. Louis	20
University of North Carolina–Chapel Hill	21
University of Texas–Austin	21
Carnegie Mellon University	21
University of Wisconsin–Madison	24
Georgia Institute of Technology	25
University of Illinois–Urbana-Champaign	25
Boston University	27
University of Southern California	28
Brown University	29
University of California–Davis	30
Ohio State University–Columbus	31
University of California–Santa Barbara	31
University of Minnesota–Twin Cities	33
University of Maryland–College Park	34
Emory University	35
University of Pittsburgh	35
Vanderbilt University	35
Pennsylvania State University–University Park	38
Michigan State University	39

University	Rank
University of California–Irvine	39
Purdue University–West Lafayette	41
Case Western Reserve University	42
University of Florida	42
Arizona State University–Tempe	44
University of Colorado–Boulder	44
University of Virginia	44
Rice University	47
University of Rochester	48
University of Arizona	48
Rutgers, The State University of New Jersey–New Brunswick	50
Indiana University–Bloomington	51
Texas A&M University–College Station	51
University of Massachusetts–Amherst	53
Tufts University	54
University of California–Santa Cruz	55
Northeastern University	55
University of Miami	57
University of California–San Francisco	57
Dartmouth College	57
Rockefeller University	60
University of Utah	60
George Washington University	60
University of California–Riverside	63
University of Iowa	64
University of Notre Dame	64
Florida State University	66
University of Texas Southwestern Medical Center–Dallas	66
Baylor College of Medicine	68
Oregon Health and Science University	69
North Carolina State University–Raleigh	70
Georgetown University	70
Virginia Tech	72
Stony Brook University–SUNY	73
University of Tennessee	74
University of Illinois–Chicago	75
Colorado State University	76
University of Kansas	77
University of South Florida	78
University of Hawaii–Manoa	79
University at Buffalo–SUNY	79

University	Rank
Washington State University	81
Brandeis University	82
University of Texas–Dallas	83
University of Delaware	83
University of Oregon	85
University of Connecticut	86
University of Missouri	87
Boston College	88
Iowa State University	89
Oregon State University	90
University of Georgia	91
University of Nebraska–Lincoln	92
University of Cincinnati	93
University of New Mexico	94
Drexel University	95
Temple University	95
University of South Carolina–Columbia	97
Wake Forest University	97
Wayne State University	99
University of Kentucky	99

Table 5.8: Global Rank of U.S. Universities That Have Appeared at Least Once in the Input 2021 Top-100 U.S. Universities Lists

University	US News	Times	QS	ARWU
Harvard University	1	3	3	1
Massachusetts Institute of Technology	2	5	1	3
Stanford University	3	2	2	2
University of California–Berkeley	4	7	30	5
Columbia University	6	17	19	8
University of Washington	8	29	72	19
California Institute of Technology	7	4	4	9
Johns Hopkins University	10	12	25	16
University of California–San Francisco	15	−−	−−	20
Yale University	11	8	17	11
University of Pennsylvania	14	13	16	15
University of California–Los Angeles	13	15	36	14
University of Chicago	15	10	9	10
Princeton University	11	9	12	6
University of Michigan–Ann Arbor	17	22	21	26
University of California–San Diego	21	33	54	18
Cornell University	22	19	18	12
Duke University	23	20	42	32
Northwestern University	24	24	29	34
New York University	29	26	35	27
Washington University in St. Louis	33	50	105	23
University of North Carolina–Chapel Hill	36	56	95	29
University of Pittsburgh	43	133	156	101
University of Texas–Austin	38	44	71	41
Ohio State University–Columbus	45	80	108	101
University of Wisconsin–Madison	41	49	65	31
University of Minnesota–Twin Cities	47	85	177	40
Icahn School of Medicine at Mount Sinai	62	−−	−−	101
Georgia Institute of Technology	66	38	80	101
University of Maryland–College Park	60	90	152	56
University of Colorado–Boulder	59	131	230	46
Boston University	57	54	110	95
University of California–Davis	66	64	112	100
University of California–Santa Barbara	56	68	152	57
University of Southern California	70	53	121	61
University of Illinois–Urbana-Champaign	60	48	82	55
Vanderbilt University	72	111	187	65

University	US News	Times	QS	ARWU
Emory University	71	85	158	101
Pennsylvania State University–University Park	75	114	101	101
University of California–Irvine	78	98	210	70
Rockefeller University	76	−−	−−	42
University of Arizona	97	124	273	101
University of Florida	107	152	162	97
Carnegie Mellon University	94	28	51	97
University of California–Santa Cruz	81	201	416	151
Michigan State University	100	105	157	101
University of Virginia	109	117	217	151
Brown University	101	61	60	101
University of Texas Southwestern Medical Center–Dallas	139	−−	−−	48
Purdue University–West Lafayette	114	94	109	86
Rutgers, The State University of New Jersey–New Brunswick	113	166	258	101
Baylor College of Medicine	132	−−	−−	151
Texas A&M University–College Station	130	197	169	151
Indiana University–Bloomington	127	140	320	101
Case Western Reserve University	142	121	162	101
University of Alabama–Birmingham	155	169	−−	301
University of Massachusetts–Amherst	142	201	270	201
Oregon Health and Science University	148	251	−−	201
University of Utah	142	201	333	101
University of Rochester	140	147	165	151
Arizona State University–Tempe	146	184	220	101
Rice University	128	124	89	101
University of California–Riverside	158	251	449	201
University of Iowa	160	201	420	201
Northeastern University	177	176	362	201
University of Colorado Anschutz Medical Campus	−−	301	−−	−−
University of Cincinnati	202	251	601	201
Stony Brook University–SUNY	176	301	373	301
Tufts University	182	155	260	151
University of Tennessee	217	301	432	201
Florida State University	195	251	456	201
North Carolina State University–Raleigh	226	301	295	201
University of Illinois–Chicago	212	251	256	201

University	US News	Times	QS	ARWU
Iowa State University	231	−−	541	301
University of Miami	236	201	291	201
University of Oregon	232	301	601	301
Dartmouth College	226	101	203	301
George Washington University	258	187	353	301
Virginia Tech	247	201	326	201
Yeshiva University	261	−−	341	701
Colorado State University	−−	−−	443	201
University of Maryland–Baltimore	271	−−	−−	301
University of Texas Health Science Center–Houston	261	−−	−−	301
University at Buffalo–SUNY	277	251	346	301
University of Kansas	284	351	383	201
University of Notre Dame	238	170	211	301
University of New Mexico	256	251	601	401
Oregon State University	277	−−	499	201
University of South Florida	312	201	581	251
Washington State University	297	301	411	301
Temple University	312	301	701	501
University of Georgia	310	−−	501	201
University of Texas–Dallas	269	301	531	301
Georgetown University	322	120	230	201
Brandeis University	224	201	446	301
University of Connecticut	332	−−	501	201
University of South Carolina–Columbia	−−	−−	571	−−
University at Albany–SUNY	297	251	751	401
University of Delaware	336	301	541	201
University of Kentucky	358	−−	601	301
Drexel University	387	351	601	301
University of Colorado–Denver	363	−−	398	201
University of Mississippi	355	−−	801	−−
Wayne State University	327	351	477	401
University of Nebraska–Lincoln	349	−−	561	201
University of Hawaii–Manoa	378	201	333	301
Syracuse University	370	301	601	601
Tulane University	440	301	414	601
Virginia Commonwealth University	402	−−	651	201
University of Missouri	−−	−−	483	201
Wake Forest University	419	251	380	401
University of Houston	399	−−	701	201

University	US News	Times	QS	ARWU
George Mason University	453	251	801	201
Rush University	467	301	−−	501
University of Texas–San Antonio	504	501	−−	501
William & Mary	564	251	651	−−
Boston College	595	301	454	401
Rensselaer Polytechnic Institute	577	501	470	401
University of California–Merced	693	301	−−	401
Illinois Institute of Technology	736	301	426	801
Howard University	761	201	651	−−
California State University–Fresno	494	−−	−−	−−
University of Alaska–Fairbanks	736	401	−−	501
Nova Southeastern University	877	251	−−	−−
Lehigh University	850	601	551	801
Clark University	−−	−−	581	−−
The New School	−−	801	561	−−
The University of Texas M. D. Anderson Cancer Center	−−	−−	67	−−
Mayo Clinic Alix School of Medicine	−−	−−	−−	101
University of Massachusetts Medical– School - Worcester	374	−−	−−	201

Table 5.9: 2021 Consensus Top-100 U.S. Universities

University	Rank
Harvard University	1
Stanford University	1
Massachusetts Institute of Technology	3
University of California–Berkeley	4
California Institute of Technology	4
Princeton University	6
University of Chicago	7
Yale University	7
Columbia University	7
Johns Hopkins University	10
University of Pennsylvania	10
University of California–Los Angeles	12
Cornell University	12
University of Michigan–Ann Arbor	14
University of Washington	14
Duke University	16
Northwestern University	17
New York University	18
University of California–San Diego	18
University of Texas–Austin	20
Washington University in St. Louis	20
Carnegie Mellon University	20
University of Wisconsin–Madison	20
University of North Carolina–Chapel Hill	24
University of Illinois–Urbana-Champaign	24
Georgia Institute of Technology	24
Boston University	27
University of Southern California	27
University of California–Davis	29
University of California–Santa Barbara	29
Brown University	29
Ohio State University–Columbus	32
University of Minnesota–Twin Cities	33
University of Maryland–College Park	34
Emory University	35
Vanderbilt University	36
Pennsylvania State University–University Park	37
University of Colorado–Boulder	37
Purdue University–West Lafayette	37

University	Rank
University of Pittsburgh	40
University of California–Irvine	40
Michigan State University	42
University of Florida	43
Rice University	43
University of Virginia	45
Case Western Reserve University	45
University of Arizona	47
Rutgers, The State University of New Jersey–New Brunswick	48
Indiana University–Bloomington	49
University of Rochester	50
Arizona State University–Tempe	51
Texas A&M University–College Station	51
Tufts University	53
Dartmouth College	53
Georgetown University	55
University of Utah	55
University of California–Santa Cruz	57
University of Massachusetts–Amherst	58
Northeastern University	59
University of Iowa	60
University of California–San Francisco	60
University of Notre Dame	60
University of Illinois–Chicago	63
University of Miami	64
Virginia Tech	65
Rockefeller University	65
Icahn School of Medicine at Mount Sinai	65
University of California–Riverside	68
Florida State University	69
University of Cincinnati	70
Oregon Health and Science University	70
University of Alabama–Birmingham	72
North Carolina State University–Raleigh	72
George Washington University	74
University of Hawaii–Manoa	75
University at Buffalo–SUNY	76
Stony Brook University–SUNY	77
University of Tennessee	78
University of Kansas	79
Brandeis University	79

University	Rank
Oregon State University	81
University of Texas–Dallas	81
Washington State University	81
University of Georgia	84
University of Connecticut	85
University of Delaware	86
University of Colorado–Denver	87
University of South Florida	87
Iowa State University	89
University of Oregon	90
University of New Mexico	91
Wake Forest University	91
University at Albany–SUNY	93
Tulane University	94
Boston College	95
Wayne State University	96
University of Nebraska–Lincoln	97
Syracuse University	98
Drexel University	99
Rensselaer Polytechnic Institute	100

Table 5.10: Global Rank of U.S. Universities That Have Appeared at Least Once in the Input 2020 Top-100 U.S. Universities Lists

University	US News	Times	QS	ARWU
Harvard University	1	7	3	1
Massachusetts Institute of Technology	2	5	1	4
Stanford University	3	4	2	2
University of California–Berkeley	4	13	28	5
Columbia University	7	14	18	7
University of Washington	10	26	68	16
California Institute of Technology	6	2	5	8
Johns Hopkins University	11	12	24	15
University of California–San Francisco	15	−−	1001	21
Yale University	12	8	17	11
University of Pennsylvania	16	11	15	19
University of California–Los Angeles	14	17	35	13
University of Chicago	13	9	10	10
Princeton University	8	6	13	6
University of Michigan–Ann Arbor	17	21	21	22
University of California–San Diego	19	31	45	18
Cornell University	23	19	14	12
Duke University	22	20	25	27
Northwestern University	24	22	31	30
New York University	28	29	39	27
Washington University in St. Louis	31	52	108	23
University of North Carolina–Chapel Hill	33	54	90	30
University of Pittsburgh	47	113	140	96
University of Texas–Austin	34	38	65	41
Ohio State University–Columbus	45	70	101	101
University of Wisconsin–Madison	37	51	56	32
University of Minnesota–Twin Cities	47	79	156	40
Icahn School of Medicine–Mount Sinai	67	−−	−−	101
Georgia Institute of Technology	62	38	72	101
University of Maryland–College Park	51	91	136	53
University of Colorado–Boulder	50	124	206	44
Boston University	51	61	98	90
University of California–Davis	−−	55	104	91
University of California–Santa Barbara	41	57	135	49
University of Southern California	69	62	129	61
University of Illinois–Urbana-Champaign	59	48	75	45
Vanderbilt University	72	116	200	62

University	US News	Times	QS	ARWU
Emory University	71	80	156	101
Pennsylvania State University–University Park	72	78	93	101
University of California–Irvine	78	96	219	69
Rockefeller University	62	−−	−−	43
University of Arizona	85	104	262	101
University of Florida	105	175	167	88
Carnegie Mellon University	82	27	48	95
University of California–Santa Cruz	76	179	367	151
Michigan State University	101	84	144	101
University of Virginia	107	107	198	151
Brown University	102	53	57	101
University of Texas Southwestern Medical Center–Dallas	146	−−	−−	−−
Purdue University–West Lafayette	114	88	111	79
Rutgers, The State University of New Jersey–New Brunswick	105	168	262	101
Baylor College of Medicine	132	−−	1001	151
Texas A&M University–College Station	134	178	189	151
Indiana University–Bloomington	136	134	312	101
Case Western Reserve University	155	119	167	101
University of Alabama–Birmingham	175	172	−−	301
University of Massachusetts–Amherst	136	201	305	151
Oregon Health and Science University	152	251	−−	201
University of Utah	139	201	353	101
University of Rochester	125	173	170	151
Arizona State University–Tempe	146	155	215	101
Rice University	108	105	85	101
University of California–Riverside	149	251	454	201
University of Iowa	161	201	421	201
Northeastern University	200	173	344	201
University of Cincinnati	196	251	561	201
Stony Brook University–SUNY	171	301	359	201
Tufts University	190	139	253	151
University of Tennessee	217	301	407	201
Florida State University	190	251	448	201
North Carolina State University–Raleigh	214	301	285	201
University of Illinois–Chicago	217	251	231	201
Iowa State University	220	351	511	201
University of Miami	235	201	271	201

University	US News	Times	QS	ARWU
University of Oregon	220	251	601	301
Dartmouth College	214	94	207	201
George Washington University	255	198	336	301
Virginia Tech	253	201	327	201
Yeshiva University	262		359	201
Colorado State University	——	401	442	201
University of Maryland–Baltimore	257			201
University at Buffalo–SUNY	266	251	340	301
University of Kansas	279	401	372	201
University of Notre Dame	223	157	210	301
University of New Mexico	253	301	601	301
Oregon State University	266	351	461	201
University of South Florida	310	201	601	201
Washington State University	290	351	400	301
Temple University	319	301	651	301
University of Georgia	290	401	474	201
University of Texas–Dallas	273	301	501	301
Georgetown University	298	102	226	201
Brandeis University	250	201	468	301
University of Connecticut	325	351	377	201
University of South Carolina–Columbia	——	401	561	301
University at Albany–SUNY	298	351	701	501
University of Delaware	311	251	491	201
University of Kentucky	349	401	601	301
Drexel University	405	401	561	301
University of Colorado–Denver	338	——	392	201
Wayne State University	309	351	484	501
University of Nebraska–Lincoln	290	401	561	201
Baylor University	353	601	801	501
University of Hawaii–Manoa	379	201	326	301
Syracuse University	346	251	581	601
Tulane University	469	301	419	601
University of Massachusetts–Boston	528	——	571	901
Virginia Commonwealth University	402	——	601	201
University of Missouri	——	401	551	151
University of Oklahoma	405	——	571	401
Wake Forest University	401	201	398	401
University of Houston	385	401	651	201
George Mason University	434	251	801	201

University	US News	Times	QS	ARWU
Rush University	443	301	−−	501
Indiana University-Purdue University–Indianapolis	315	−−	651	301
William & Mary	497	201	601	901
Colorado School of Mines	503	301		601
Boston College	536	301	432	501
Rensselaer Polytechnic Institute	519	401	383	401
University of California–Merced	667	351	−−	401
Illinois Institute of Technology	682	−−	421	−−
Howard University	774	201	581	901
University of Maryland–Baltimore County	660	601	460	601
University of Alaska–Fairbanks	676	351	−−	601
University of Denver	727	351	651	901
Nova Southeastern University	928	301	−−	−−
Lehigh University	799	601	551	701
Clark University	−−	351	531	−−
The New School	−−	801	561	−−
The University of Texas M. D. Anderson Cancer Center	−−	−−	−−	67
University of Massachusetts Medical– School - Worcester	374	−−		151

Table 5.11: 2020 Consensus Top-100 U.S. Universities

University	Rank
Harvard University	1
Stanford University	1
Massachusetts Institute of Technology	1
California Institute of Technology	4
Princeton University	5
University of California–Berkeley	5
Columbia University	7
University of Chicago	7
Yale University	7
University of Pennsylvania	10
Johns Hopkins University	10
Cornell University	10
University of California–Los Angeles	13
University of Michigan–Ann Arbor	14
University of Washington	14
Duke University	16
University of California–San Diego	16
Northwestern University	18
New York University	19
University of Wisconsin–Madison	20
University of Texas–Austin	20
Washington University in St. Louis	20
Carnegie Mellon University	20
Georgia Institute of Technology	24
University of Illinois–Urbana-Champaign	24
Brown University	26
University of North Carolina–Chapel Hill	26
University of California–Santa Barbara	28
Boston University	29
Ohio State University–Columbus	30
University of Minnesota–Twin Cities	31
University of Southern California	31
University of California–Davis	31
Pennsylvania State University–University Park	34
Purdue University–West Lafayette	35
University of Maryland–College Park	35
University of Pittsburgh	37
Emory University	38
Michigan State University	39

University	Rank
University of Colorado–Boulder	39
Vanderbilt University	39
University of California–Irvine	42
University of Arizona	43
University of Florida	43
Rice University	45
Case Western Reserve University	46
University of Virginia	46
Rutgers, The State University of New Jersey–New Brunswick	48
Indiana University–Bloomington	49
University of Rochester	49
Texas A&M University–College Station	51
Arizona State University–Tempe	51
Tufts University	53
Dartmouth College	53
University of California–Santa Cruz	55
Georgetown University	55
University of Massachusetts–Amherst	57
Northeastern University	58
University of Utah	58
University of Notre Dame	58
University of Iowa	61
University of Illinois–Chicago	62
University of Miami	63
Virginia Tech	64
University of California–Riverside	65
University of California–San Francisco	65
Rockefeller University	67
Oregon Health and Science University	68
Stony Brook University–SUNY	69
Florida State University	70
University of Cincinnati	71
North Carolina State University–Raleigh	72
University of Alabama–Birmingham	72
George Washington University	74
University of Tennessee	74
University of Hawaii–Manoa	76
Brandeis University	76
Iowa State University	76
Yeshiva University	79
University at Buffalo–SUNY	79

University	Rank
Oregon State University	81
University of Kansas	82
University of South Florida	83
University of Delaware	84
University of Connecticut	85
University of Oregon	85
University of New Mexico	87
University of Texas–Dallas	87
Washington State University	89
University of Georgia	90
University of Colorado–Denver	91
Wake Forest University	92
Wayne State University	93
University of Missouri	94
University of Nebraska–Lincoln	95
Syracuse University	96
Colorado State University	96
Temple University	96
University of Kentucky	99
Drexel University	99

Table 5.12: $K^{(p)}$ Distance, # of Full Rank Reversals, and # of and Partial Rank Reversals, Respectively, Between the Inputs Lists and Consensus, For 2022

	USNEWS	TIMES	QS	ARWU	Consensus
USNEWS	0	1,432; 1,209; 206	1,414; 1,296; 26	938.5; 661; 483	718; 670; 54
TIMES	1,432; 1,209; 206	0	1,112; 859; 200	1,736; 1257; 578	997; 745; 232
QS	1,414; 1,296; 26	1,112; 859; 200	0	1,658.5; 1,283; 479	867.5; 763, 53
ARWU	938.5; 661; 483	1,736; 1257; 578	1,658.5; 1,283; 479	0	972; 659; 516
Consensus	718, 670, 54	997, 745, 232	867.5, 763, 53	972, 659, 516	0

Table 5.13: $K^{(p)}$ Distance, # of Full Rank Reversals, and # of and Partial Rank Reversals, Respectively, Between the Inputs Lists and Consensus, For 2021

	USNEWS	TIMES	QS	ARWU	Consensus
USNEWS	0	1,210; 1,004; 202	1,370; 1269; 21	850; 564; 516	662; 606; 56
TIMES	1,210; 1,004; 202	0	1,006.5; 801; 201	1,583.5; 1,115; 595	789.5; 585; 227
QS	1,370; 1,269; 21	1,006.5; 801; 201	0	1,504; 1,153; 520	895; 791; 52
ARWU	850; 564; 516	1,583.5; 1,115; 595	1,504; 1,153; 520	0	949; 626; 556
Consensus	662; 606; 56	789.5; 585; 22	895; 791; 52	949; 626; 556	0

Table 5.14: $K^{(p)}$ Distance, # of Full Rank Reversals, and # of and Partial Rank Reversals, Respectively, Between the Inputs Lists and Consensus, For 2020

	USNEWS	TIMES	QS	ARWU	Consensus
USNEWS	0	1,252; 1,082; 184	1,385.5; 1251; 29	944; 621; 574	633; 579; 66
TIMES	1,252; 1,082; 184	0	981; 775, 172	1,555; 1,114; 610	837.5; 641; 211
QS	1,385.5; 1251; 29	981, 775; 172	0	1,465.5; 1,067; 557	935.5; 784; 63
ARWU	944; 621; 574	1,555; 1,114; 610	1,465.5; 1,067; 557	0	1,020.5; 658; 593
Consensus	633; 579; 66	837.5; 642; 211	933.5; 783; 63	1,020.5; 659; 593	0

5.9 Concluding Remarks

This chapter studies the top-k list aggregation problem. It presents a binary non-linear and four mixed-integer linear programming formulations. Furthermore, it studies the strength of the four mixed-integer linear programming formulations using polyhedral analysis. Our findings show that the presented formulations can be ordered based on the strength of their LP relaxations. The strongest formulation is induced by a novel set of preference cycle-prevention constraints tailored to the specific structure of the top-k list aggregation problem introduced herein. Furthermore, it studies the theoretical insights for setting the associated fixed penalty parameter and the implications thereof, and it introduces various heuristics and a data reduction technique for improving their run time on certain large-scale instances. In experiments conducted on synthetic instances ranging from top-10 to top-1,000 lists, the proposed IGH and H-SN-SA methods had the best performance; IGH produced the highest-quality solutions among the heuristic methods, while H-SN-SA yielded competitive solutions in much less time. Based on the conducted experiments, H-SN-SA is the only technique that can be applied on massive instances. Data reduction proved to be a valuable technique, especially for exact methods. Based on a broad spectrum of generated data sets, implicit reduction performed much better, especially on large-scale instances. Finally, this chapter studies the university rankings problem by aggregating the top-100 U.S. universities, as ranked by four different services.

Chapter 6

GENERAL DISCUSSION AND CONCLUSIONS

Rank aggregation is increasingly being utilized in various fields, bringing new challenges to this well-studied problem. In particular, recent applications in artificial intelligence have raised new challenges such as high-dimensionality and ubiquity of partial information. Another challenge comes perhaps from the group decision-making applications where it is not possible for humans to express their preference in a strict manner over more than a handful of items.

This dissertation revolves around robust mathematical frameworks to overcome some of the existing challenges in the literature. It seeks to overcome the difficulty of solving high-dimensional instances by proposing Approximate Condorcet Partitioning (ACP). More specifically, we formalize the concept of the finest-Condorcet partition, specify its requirements, prove its uniqueness, and derive an efficient algorithm for its construction. However, the finest-Condorcet partition often yields a few large subsets which may be too difficult to solve with exact methods. To overcome this issue, ACP tries to further break down the larger subsets. ACP is accompanied by instance-specific approximation factors. Although the approximation factors are not constant, they were lower than the best-known approximation factors. Future research will explore other approximate partitioning schemes. Additionally, investigating whether the run time of ACP can be decreased by using specialized solution techniques such as constraint relaxation can be another direction for future research.

Moreover, this work utilizes the generalized Kendall-tau distance (Fagin *et al.*, 2004) to introduce and study Rank-Agg, which contains Kemeny-Agg as a special case. Specifically, it introduces various exact and inexact solution techniques.

Additionally, this dissertation broadens the applicability of Condorcet extensions by introducing a new social choice property (GXCC), which includes XCC and NXCC as special cases. Using GXCC, many instances of this NP-hard problem can be decomposed into smaller subproblems, while proving that solving the subproblems independently and concatenating the results still provides an optimal solution to the original problem. Conducted computations demonstrates that GXCC can be effective in reducing the run time of exact methods and improving the solution quality of inexact methods. Future research will explore the development of additional exact, approximate, and heuristic algorithms for Rank-Agg.

Finally, this work explores the top-k list aggregation problem. The top-k list aggregation problem can be divided based on whether the output ranking is considered a full list or another top-k list. Even though the former type of top-k list aggregation problem has been studied in the literature, there are little to no studies on the latter type of top-k list aggregation problem, which is studied herein. This work derives the first exact mathematical formulations for this problem and compares the strength of the proposed formulations from a polyhedral point of view. Additionally, it introduces various heuristics and a data reduction technique for improving their run time on certain large-scale instances. Even though there are various distance measures between top-k lists, this work focuses on the generalized Kendall-tau distance (Fagin *et al.*, 2003); however, the mathematical formulations can be used by modifying their objective functions accordingly. Moreover, their results on comparing proposed formulations hold regardless of the distance measure. This work also derives theoretical insights for setting the associated fixed penalty parameter and the implications thereof. Future research will explore additional exact and heuristic algorithms and data reduction techniques.

In spite of the fact that this dissertation addresses some of the existing challenges,

various others highlighted in the literature remain. An existing challenge is finding all optimal solutions. Rank aggregation problems may return many alternative optimal solutions (Pedings *et al.*, 2012); in order to be fair to all candidates, it is very important to identify them all. To the best of our knowledge, the only solution technique that is capable of recovering all optimal solutions is the branch & bound method; however, this method is capable of handling instances with less than 20 items (Yoo *et al.*, 2020). This topic becomes very important in voting theory and in situations where the aggregate ranking has strong implications such as the university rankings studied in Chapter 5.

Most importantly, the topic of incomplete rank aggregation requires significantly more attention. There are numerous situations in group decision-makings context where judges are not able to express their preferences over all items to produce a complete ranking; in these circumstances, usually judges rank only a subset of items, which may differ in size. The reasons include practicality, feasibility, and judiciousness (Moreno-Centeno and Escobedo, 2016). Examples of this happening are the National Science Foundation (NSF) proposal review process (Moreno-Centeno and Escobedo, 2016), student paper competition (Escobedo *et al.*, 2022), and corporate project selection (Yoo *et al.*, 2020). Another example is the case where judges are asked to rank movies, songs, books, brands, etc (Xiao *et al.*, 2021). Aside from the frequent settings where judges are not able to rank all items, one of the main advantages of using incomplete rankings is to avoid overranking fatigue and bias (Moreno-Centeno and Escobedo, 2016). Due to the high frequency of this phenomenon, developing adequate aggregation functions capable of aggregating incomplete rankings is pivotal. Additionally, developing specialized partitioning techniques for incomplete rank aggregation is another promising research direction as these techniques have been shown, partly in this dissertation, to be very powerful at solving large instances.

"Alaska better elections implementation", https://www.elections.alaska.gov/RCV.php, retrieved August 25, 2022 (2022). 1.1

"Best global universities", https://www.usnews.com/education/best-global-universities, retrieved August 25, 2022 (2022). 5.8

"Best global universities", https://www.timeshighereducation.com/world-university-rankings/2022, retrieved August 25, 2022 (2022). 5.8

"Best global universities", https://www.shanghairanking.com/rankings, retrieved August 25, 2022 (2022a). 5.8

"Columbia whistleblower on exposing college rankings: 'they are worthless"', https://www.theguardian.com/us-news/2022/sep/16/columbia-whistleblower-us-news-rankings-michael-thaddeus, retrieved August 25, 2022 (2022). 1.2

"Qs world university rankings", https://www.topuniversities.com/qs-world-university-rankings, retrieved August 25, 2022 (2022). 5.8

"Ranked choice voting in maine", https://legislature.maine.gov/lawlibrary/ranked-choice-voting-in-maine/9509, retrieved August 25, 2022 (2022). 1.1

"Shanghairanking's academic ranking of world universities methodology 2022", https://www.shanghairanking.com/methodology/arwu/2022, retrieved August 18, 2022 (2022b). 5.8

"Where ranked choice voting is used", http://www.fairvote.org/where_is_ranked_choice_voting_used, retrieved August 25, 2022 (2022). 1.1

Ailon, N., "Aggregation of partial rankings, p-ratings and top-m lists", Algorithmica **57**, 2, 284–300 (2010). 1.1, 1.2, 2.3.2, 5.2, 5.2, 5.2

Ailon, N., M. Charikar and A. Newman, "Aggregating inconsistent information: ranking and clustering", Journal of the ACM (JACM) **55**, 5, 1–27 (2008). 1.1, 1.2, 2.3.2, 5.5

Akbari, S. and A. R. Escobedo, "Lower bounds on kemeny rank aggregation with non-strict rankings", in "2021 IEEE Symposium Series on Computational Intelligence (SSCI)", pp. 1–8 (IEEE, 2021). 2, 4.3.1

Akbari, S. and A. R. Escobedo, "Approximate condorcet partitioning: Solving very large rank aggregation problems at scale", under review (2022a). 1

Akbari, S. and A. R. Escobedo, "Beyond kemeny aggregation: Theoretical and computational insights for robust ranking aggregation", under review (2022b). 3

Akbari, S. and A. R. Escobedo, "Top-k list aggregation: Mathematical formulations and polyhedral comparisons", in "To appear in International Symposium on Combinatorial Optimization(ISCO), Lecture Notes in Computer Science", (2022c). 4

Aledo, J. A., J. A. Gámez and D. Molina, "Tackling the supervised label ranking problem by bagging weak learners", Information Fusion **35**, 38–50 (2017). 1.1

Aledo, J. A., J. A. Gámez and D. Molina, "Approaching the rank aggregation problem by local search-based metaheuristics", Journal of Computational and Applied Mathematics **354**, 445–456 (2019). 1.1

Aledo, J. A., J. A. Gámez, D. Molina and A. Rosete, "Consensus-based journal rankings: A complementary tool for bibliometric evaluation", Journal of the Association for Information Science and Technology **69**, 7, 936–948 (2018). 1.1

Aledo, J. A., J. A. Gámez and A. Rosete, "A highly scalable algorithm for weak rankings aggregation", Information Sciences **570**, 144–171 (2021). 3.1, 3.2.1

Amodio, S., A. D'Ambrosio and R. Siciliano, "Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the kemeny axiomatic approach", European Journal of Operational Research **249**, 2, 667–676 (2016). 1.1

Andrieu, P., B. Brancotte, L. Bulteau, S. Cohen-Boulakia, A. Denise, A. Pierrot and S. Vialette, "Efficient, robust and effective rank aggregation for massive biological datasets", Future Generation Computer Systems (2021). 1.2, 4.3, 22

Arrow, K. J., *Social choice and individual values* (New Haven, CT: Cowles Foundation, 1951). 1.1

Azzini, I. and G. Munda, "A new approach for identifying the kemeny median ranking", European Journal of Operational Research **281**, 2, 388–401 (2020). 1.1

Balas, E., "Projection, lifting and extended formulation in integer and combinatorial optimization", Annals of Operations Research **140**, 1, 125–161 (2005). 18

Bartholdi, J., C. A. Tovey and M. A. Trick, "Voting schemes for which it can be difficult to tell who won the election", Social Choice and welfare **6**, 2, 157–165 (1989). 1.1, 4.3.1

Benítez-Fernández, A. and F. Ruiz, "A meta-goal programming approach to cardinal preferences aggregation in multicriteria problems", Omega **94**, 102045 (2020). 1.1

Betzler, N., R. Bredereck and R. Niedermeier, "Theoretical and empirical evaluation of data reduction for exact kemeny rank aggregation", Autonomous Agents and Multi-Agent Systems **28**, 5, 721–748 (2014). 1.1, 1.2, 2.3.2, 3.1

Bolón-Canedo, V. and A. Alonso-Betanzos, "Ensembles for feature selection: A review and future trends", Information Fusion **52**, 1–12 (2019). 1.1

Borda, J. d., "Mémoire sur les élections au scrutin", Histoire de l'Academie Royale des Sciences pour 1781 (Paris, 1784) (1784). 1.1

Bowman, N. A. and M. N. Bastedo, "Getting on the front page: Organizational reputation, status signals, and the impact of us news and world report on student decisions", Research in Higher Education **50**, 5, 415–436 (2009). 5.8

Bradley, R. A. and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons", Biometrika **39**, 3/4, 324–345 (1952). 1.2

Brancotte, B., B. Yang, G. Blin, S. Cohen-Boulakia, A. Denise and S. Hamel, "Rank aggregation with ties: Experiments and analysis", Proceedings of the VLDB Endowment (PVLDB) **8**, 11, 1202–1213 (2015). 4.3, 4.3.1, 4.6

Brandt, F., V. Conitzer, U. Endriss, J. Lang and A. D. Procaccia, *Handbook of computational social choice* (Cambridge University Press, 2016). 1.1, 2.3, 2.4, 5.2

Braverman, M. and E. Mossel, "Sorting from noisy information", arXiv preprint arXiv:0910.1191 (2009). 2.2

Çakır, M. P., C. Acartürk, O. Alaşehir and C. Çilingir, "A comparative analysis of global and national university ranking systems", Scientometrics **103**, 3, 813–848 (2015). 5.8

Cascaro, R. J., B. D. Gerardo and R. P. Medina, "Aggregating filter feature selection methods to enhance multiclass text classification", in "Proceedings of the 2019 7th International Conference on Information Technology: IoT and Smart City", pp. 80–84 (2019). 1.1

Chatterjee, S., A. Mukhopadhyay and M. Bhattacharyya, "A weighted rank aggregation approach towards crowd opinion analysis", Knowledge-Based Systems **149**, 47–60 (2018). 1.1

Chen, S., J. Liu, H. Wang and J. C. Augusto, "Ordering based decision making–a survey", Information Fusion **14**, 4, 521–531 (2013). 1.1

Chen, Y., J. Fan, C. Ma and K. Wang, "Spectral method and regularized mle are both optimal for top-k ranking", Annals of statistics **47**, 4, 2204 (2019). 1.2

Chen, Y., H. Guo, Y. Zhang, C. Ma, R. Tang, J. Li and I. King, "Learning binarized graph representations with multi-faceted quantization reinforcement for top-k recommendation", arXiv preprint arXiv:2206.02115 (2022). 1.1

Chierichetti, F., A. Dasgupta, S. Haddadan, R. Kumar and S. Lattanzi, "Mallows models for top-k lists", in "Advances in Neural Information Processing Systems", pp. 4382–4392 (2018). 1.1, 5.2.1, 5.7.1, 5.7.1

Chierichetti, F., A. Dasgupta, R. Kumar and S. Lattanzi, "On reconstructing a hidden permutation", in "Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)", (Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014). 2.2

Cohen-Boulakia, S., A. Denise and S. Hamel, "Using medians to generate consensus rankings for biological data", in "International Conference on Scientific and Statistical Database Management", pp. 73–90 (Springer, 2011). (document), 1.2, 3.2.2, 4.6, 4.6, 4.2, 4.6, 4.5

Collas, F. and E. Irurozki, "Concentric mixtures of mallows models for top-k rankings: sampling and identifiability", in "International Conference on Machine Learning", pp. 2079–2088 (PMLR, 2021). 1.2

Collier, J. H. and A. S. Konagurthu, "An information measure for comparing top k lists", in "2014 IEEE 10th international conference on e-science", vol. 1, pp. 127–134 (IEEE, 2014). 5.2

Conitzer, V., A. Davenport and J. Kalagnanam, "Improved bounds for computing kemeny rankings", in "AAAI", vol. 6, pp. 620–626 (2006). 1.1, 1.2, 3.3.2, 3.3.2, 3.3.2, 3.3.2, 3.3.3, 4.3.1

Cook, W. D., "Distance-based and ad hoc consensus models in ordinal preference ranking", European Journal of operational research **172**, 2, 369–385 (2006). 1.1, 5.3

Cook, W. D., T. Raviv and A. J. Richardson, "Aggregating incomplete lists of journal rankings: An application to academic accounting journals", Accounting perspectives **9**, 3, 217–235 (2010). 1.1

Copeland, A. H., "A reasonable social welfare function", Tech. rep., mimeo, 1951. University of Michigan (1951). 1.1

Coppersmith, D., L. K. Fleischer and A. Rurda, "Ordering by weighted number of wins gives a good ranking for weighted tournaments", ACM Transactions on Algorithms (TALG) **6**, 3, 1–13 (2010). 1.1

Critchlow, D. E., *Metric methods for analyzing partially ranked data*, vol. 34 (Springer Science & Business Media, 2012). 5.2

Dantzig, G., R. Fulkerson and S. Johnson, "Solution of a large-scale traveling-salesman problem", Journal of the operations research society of America **2**, 4, 393–410 (1954). 3.3.3

Dantzig, G. B., D. R. Fulkerson and S. M. Johnson, "On a linear-programming, combinatorial approach to the traveling-salesman problem", Operations Research **7**, 1, 58–66 (1959). 3.3.3

Davenport, A. and J. Kalagnanam, "A computational study of the kemeny rule for preference aggregation", in "AAAI", vol. 4, pp. 697–702 (2004). 1.1, 3.3.1, 4.3.1

Desarkar, M. S., S. Sarkar and P. Mitra, "Preference relations based unsupervised rank aggregation for metasearch", Expert Systems with Applications **49**, 86–98 (2016). 1.1

Diaconis, P., "Group representations in probability and statistics", Lecture notes-monograph series **11**, i–192 (1988). 1.1

Diaconis, P. and R. L. Graham, "Spearman's footrule as a measure of disarray", Journal of the Royal Statistical Society: Series B (Methodological) **39**, 2, 262–268 (1977). 1.1, 2.2

Dimitrakopoulos, C., S. K. Hindupur, L. Häfliger, J. Behr, H. Montazeri, M. N. Hall and N. Beerenwinkel, "Network-based integration of multi-omics data for prioritizing cancer genes", Bioinformatics **34**, 14, 2441–2448 (2018). 1.1

Ding, J., D. Han, J. Dezert and Y. Yang, "A new hierarchical ranking aggregation method", Information Sciences **453**, 168–185 (2018a). 1.1

Ding, J., D. Han and Y. Yang, "Iterative ranking aggregation using quality improvement of subgroup ranking", European Journal of Operational Research **268**, 2, 596–612 (2018b). 1.1

Dodgson, C., "A method of taking votes on more than two issues", The theory of committees and elections (1876). 2.4

Doignon, J.-P., A. Pekeč and M. Regenwetter, "The repeated insertion model for rankings: Missing link between two subset choice models", Psychometrika **69**, 1, 33–54 (2004). 3.3.4

Dwork, C., R. Kumar, M. Naor and D. Sivakumar, "Rank aggregation methods for the web", in "Proceedings of the 10th international conference on World Wide Web", pp. 613–622 (2001). 1.1, 1.2, 2.3.2, 3.2.2, 4.3.1, 5.2, 5.2, 5.2.1, 5.5, 5

D'Ambrosio, A., C. Iorio, M. Staiano and R. Siciliano, "Median constrained bucket order rank aggregation", Computational Statistics **34**, 2, 787–802 (2019). 1.2

D'Ambrosio, A., G. Mazzeo, C. Iorio and R. Siciliano, "A differential evolution algorithm for finding the median ranking under the kemeny axiomatic approach", Computers & Operations Research **82**, 126–138 (2017). 2.2

Emond, E. J. and D. W. Mason, "A new rank correlation coefficient with application to the consensus ranking problem", Journal of Multi-Criteria Decision Analysis **11**, 1, 17–28 (2002). 1.1, 1.2, 2.3.1, 2.3.1, 2.3.1, 4.3.1

Escobedo, A. R., E. Moreno-Centeno and R. Yasmin, "An axiomatic distance methodology for aggregating multimodal evaluations", Information Sciences **590**, 322–345 (2022). 6

Fagin, R., R. Kumar, M. Mahdian, D. Sivakumar and E. Vee, "Comparing and aggregating rankings with ties", in "Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems", pp. 47–58 (2004). 1.1, 1.2, 2.2, 4.1, 4.2, 4.2, 4.5, 6

Fagin, R., R. Kumar, M. Mahdian, D. Sivakumar and E. Vee, "Comparing partial rankings", SIAM Journal on Discrete Mathematics **20**, 3, 628–648 (2006). 1.1

Fagin, R., R. Kumar and D. Sivakumar, "Comparing top k lists", SIAM Journal on discrete mathematics **17**, 1, 134–160 (2003). 1.1, 1.2, 2.2, 5.2, 5.2, 5.2, 5.2.1, 5.2.1, 5.3.1, 5.3.1, 5.5, 6

Figueira, J. R., V. Mousseau and B. Roy, "Electre methods", in "Multiple criteria decision analysis", pp. 155–185 (Springer, 2016). 1.1

Fiorini, S. and P. C. Fishburn, "Weak order polytopes", Discrete mathematics **275**, 1-3, 111–127 (2004). 4.3.1

Galdi, P., M. Fratello, F. Trojsi, A. Russo, G. Tedeschi, R. Tagliaferri and F. Esposito, "Stochastic rank aggregation for the identification of functional neuromarkers", Neuroinformatics **17**, 4, 479–496 (2019). 1.1

Gass, S. I., "Tournaments, transitivity and pairwise comparison matrices", Journal of the Operational Research Society **49**, 6, 616–624 (1998). 3.3.2

Gionis, A., H. Mannila, K. Puolamäki and A. Ukkonen, "Algorithms for discovering bucket orders from data", in "Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining", pp. 561–566 (2006). 2.3.2

Glover, F. and E. Woolsey, "Converting the 0-1 polynomial programming problem to a 0-1 linear program", Operations research **22**, 1, 180–182 (1974). 4.3.1, 5.3

Goodman, L. A. and W. H. Kruskal, "Measures of association for cross classifications. ii: Further discussion and references", Journal of the American Statistical Association **54**, 285, 123–163 (1959). 5.2

Grötschel, M., M. Jünger and G. Reinelt, "A cutting plane algorithm for the linear ordering problem", Operations research **32**, 6, 1195–1220 (1984). 4.3.1

Hare, T., *A treatise on the election of representatives, parliamentary and municipal* (Longman, Green, Longman, & Roberts, 1861). 1.1

Hsu, D. F. and I. Taksa, "Comparing rank and score combination methods for data fusion in information retrieval", Information retrieval **8**, 3, 449–480 (2005). 1.1

Hu, B., C. Shi, W. X. Zhao and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model", in "Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining", pp. 1531–1540 (2018). 1.1

Jurman, G., S. Riccadonna, R. Visintainer and C. Furlanello, "Canberra distance on ranked lists", in "Proceedings of advances in ranking NIPS 09 workshop", pp. 22–27 (Citeseer, 2009). 5.2

Kabra, A. and A. Agarwal, "Personalized and dynamic top-k recommendation system using context aware deep reinforcement learning", in "2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)", pp. 238–247 (IEEE, 2021). 1.1

Kemeny, J. G. and L. Snell, "Preference ranking: an axiomatic approach", Mathematical models in the social sciences pp. 9–23 (1962). 1.1, 1.2, 5, 2.3.1

Kemmer, R., Y. Yoo, A. Escobedo and R. Maciejewski, "Enhancing collective estimates by aggregating cardinal and ordinal inputs", Proceedings of the AAAI Conference on Human Computation and Crowdsourcing **8**, 1, 73–82 (2020). 1.1

Kendall, M. G., "A new measure of rank correlation", Biometrika **30**, 1/2, 81–93 (1938). 1.1, 6

Kendall, M. G., "Partial rank correlation", Biometrika **32**, 3/4, 277–283 (1942). 1.2

Kendall, M. G., "The treatment of ties in ranking problems", Biometrika **33**, 3, 239–251 (1945). 1.1

Kenyon-Mathieu, C. and W. Schudy, "How to rank with few errors", in "Proceedings of the thirty-ninth annual ACM symposium on Theory of computing", pp. 95–103 (2007). 2.3.2, 3.3.2

Klementiev, A., D. Roth and K. Small, "Unsupervised rank aggregation with distance-based models", in "Proceedings of the 25th international conference on Machine learning", pp. 472–479 (2008). 1.1

Laslier, J.-F., *Tournament solutions and majority voting*, no. 7 (Springer, 1997). 2.1, 3.1.1

Laura, L., "Qs world university rankings methodology: Using rankings to start your university search", URL `https://www.topuniversities.com/qs-world-university-rankings/methodology` (2022). 5.8

Lee, H., S. Cho, Y. Jang, J. Kim and H. Woo, "Differentiable ranking metric using relaxed sorting for top-k recommendation", IEEE Access (2021). 1.1

Li, X., X. Wang and G. Xiao, "A comparative study of rank aggregation methods for partial and top ranked lists in genomic applications", Briefings in bioinformatics **20**, 1, 178–189 (2019). 1.1

Liao, H. and X. Wu, "Dnma: A double normalization-based multiple aggregation method for multi-expert multi-criteria decision making", Omega **94**, 102058 (2020). 1.1

Liu, N., Z. Xu, X.-J. Zeng and P. Ren, "An agglomerative hierarchical clustering algorithm for linear ordinal rankings", Information Sciences **557**, 170–193 (2021). 3.2.1

Long, X. and T. Suel, "Optimized query execution in large search engines with global page ordering", in "Proceedings 2003 VLDB Conference", pp. 129–140 (Elsevier, 2003). 1.1

Loreggia, A., N. Mattei, F. Rossi and K. B. Venable, "A notion of distance between cp-nets", in "Proc. of AAMAS", pp. 955–963 (2018). 4.3

Lu, J. and J. Callan, "Federated search of text-based digital libraries in hierarchical peer-to-peer networks", in "European Conference on Information Retrieval", pp. 52–66 (Springer, 2005). 1.1

Luce, R. D., *Individual choice behavior: A theoretical analysis* (Courier Corporation, 2012). 1.2

Ma, C., L. Ma, Y. Zhang, H. Wu, X. Liu and M. Coates, "Knowledge-enhanced top-k recommendation in poincaré ball", in "Proceedings of the AAAI Conference on Artificial Intelligence", vol. 35, pp. 4285–4293 (2021). 1.1

Mackenzie, J. and A. Moffat, "Examining the additivity of top-k query processing innovations", in "Proceedings of the 29th ACM International Conference on Information & Knowledge Management", pp. 1085–1094 (2020). 1.1

Mallows, C. L., "Non-null ranking models. i", Biometrika **44**, 1/2, 114–130 (1957). 1.2, 2.2

Marbach, D., J. C. Costello, R. Küffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, M. Kellis, J. J. Collins and G. Stolovitzky, "Wisdom of crowds for robust gene network inference", Nature methods **9**, 8, 796–804 (2012). 1.1, 1.2

Marden, J. I., *Analyzing and Modeling Rank Data* (CRC Press, 2014). 2.2

Marquis de Condorcet, M. J. A., *Essai sur l'application de l'analyse a la probabilite des decisions: rendues a la pluralite de voix* (De l'Imprimerie royale, 1785). 2.4

Mattei, N. and T. Walsh, "Preflib: A library for preferences http://www. preflib. org", in "International Conference on Algorithmic DecisionTheory", pp. 259–270 (Springer, 2013). 1.1, 1.2, 3.2.2, 4.6

Mehta, P. and P. Majumder, "Improving sentence extraction through rank aggregation", in "From Extractive to Abstractive Summarization: A Journey", pp. 49–68 (Springer, 2019). 1.1

Milosz, R. and S. Hamel, "Exploring the median of permutations problem", Journal of Discrete Algorithms **52**, 92–111 (2018). 3.3.2, 3.3.2, 3.3.2, 4.3.1

Milosz, R. and S. Hamel, "Space reduction constraints for the median of permutations problem", Discrete Applied Mathematics **280**, 201–213 (2020). 1.1

Mohammadi, M. and J. Rezaei, "Ensemble ranking: Aggregation of rankings produced by different multi-criteria decision-making methods", Omega **96**, 102254 (2020). 1.1

Moreno-Centeno, E. and A. R. Escobedo, "Axiomatic aggregation of incomplete rankings", IIE Transactions **48**, 6, 475–488 (2016). 1.1, 6

Morse, R. and A. Castonguay, "How u.s. news calculated the best global universities rankings", URL `https://www.usnews.com/education/best-global-universities/articles/methodology` (2021). 5.8

Nápoles, G., R. Falcon, Z. Dikopoulou, E. Papageorgiou, R. Bello and K. Vanhoof, "Weighted aggregation of partial rankings using ant colony optimization", Neurocomputing **250**, 109–120 (2017). 1.2, 5.2

Nemhauser, G. L. and G. Sigismondi, "A strong cutting plane/branch-and-bound algorithm for node packing", Journal of the Operational Research Society **43**, 5, 443–457 (1992). 3.3.2, 3.3.2

Oliveira, S. E., V. Diniz, A. Lacerda, L. Merschmanm and G. L. Pappa, "Is rank aggregation effective in recommender systems? an experimental analysis", ACM Transactions on Intelligent Systems and Technology (TIST) **11**, 2, 1–26 (2020). 1.1, 1.2

Onan, A. and S. Korukoğlu, "A feature selection model based on genetic rank aggregation for text sentiment classification", Journal of Information Science **43**, 1, 25–38 (2017). 1.1

Park, C., D. Kim, J. Oh and H. Yu, "Improving top-k recommendation with truster and trustee relationship in user trust network", Information Sciences **374**, 100–114 (2016). 1.1

Pedings, K. E., A. N. Langville and Y. Yamamoto, "A minimum violations ranking method", Optimization and engineering **13**, 2, 349–370 (2012). 1.1, 3.3.3, 4.3.2, 6

Pedroche, F. and J. A. Conejero, "Corrected evolutive kendall's τ coefficients for incomplete rankings with ties: Application to case of spotify lists", Mathematics **8**, 10, 1828 (2020). 1.1

Peng, Y., G. Kou, G. Wang and Y. Shi, "Famcdm: A fusion approach of mcdm methods to rank multiclass classification algorithms", Omega **39**, 6, 677–689 (2011). 1.1

Plackett, R. L., "The analysis of permutations", Journal of the Royal Statistical Society: Series C (Applied Statistics) **24**, 2, 193–202 (1975). 1.2

Puerta, J. M., J. A. Aledo, J. A. Gámez and J. D. Laborda, "Efficient and accurate structural fusion of bayesian networks", Information Fusion **66**, 155–169 (2021). 1.1

Quillet, A., C. Saad, G. Ferry, Y. Anouar, N. Vergne, T. Lecroq and C. Dubessy, "Improving bioinformatics prediction of microrna targets by ranks aggregation", Frontiers in Genetics **10**, 1330 (2020). 1.1

Rico, N., C. R. Vela and I. Díaz, "Reducing the time required to find the kemeny ranking by exploiting a necessary condition for being a winner", European Journal of Operational Research (2022). 1.1, 1.2

Ross, D., "World university rankings 2022: methodology", URL `https://www.timeshighereducation.com/world-university-rankings/world-university-rankings-2022-methodology` (2021). 5.8

Rowland, M., S. Omidshafiei, K. Tuyls, J. Perolat, M. Valko, G. Piliouras and R. Munos, "Multiagent evaluation under incomplete information", arXiv preprint arXiv:1909.09849 (2019). 4.3

Saaty, T. L., "A scaling method for priorities in hierarchical structures", Journal of mathematical psychology **15**, 3, 234–281 (1977). 1.1

Saaty, T. L., *Decision making for leaders: the analytic hierarchy process for decisions in a complex world* (RWS publications, 2001). 1.1

Sarkar, C., S. Cooley and J. Srivastava, "Robust feature selection technique using rank aggregation", Applied Artificial Intelligence **28**, 3, 243–257 (2014). 1.1

Schalekamp, F. and A. v. Zuylen, "Rank aggregation: Together we're strong", in "2009 Proceedings of the Eleventh Workshop on Algorithm Engineering and Experiments (ALENEX)", pp. 38–51 (SIAM, 2009). 4.4.1

Shanbhag, A., H. Pirk and S. Madden, "Efficient top-k query processing on massively parallel hardware", in "Proceedings of the 2018 International Conference on Management of Data", pp. 1557–1570 (2018). 1.1

Song, D., D. A. Meyer and D. Tao, "Top-k link recommendation in social networks", in "2015 IEEE International Conference on Data Mining", pp. 389–398 (IEEE, 2015). 1.1

Tang, H., G. Zhao, Y. Wu and X. Qian, "Multisample-based contrastive loss for top-k recommendation", IEEE Transactions on Multimedia (2021). 1.1

Tavanaei, A., R. Gottumukkalay, A. S. Maida and V. V. Raghavan, "Unsupervised learning to rank aggregation using parameterized function optimization", in "2018 International Joint Conference on Neural Networks (IJCNN)", pp. 1–8 (IEEE, 2018). 1.1, 5.8

Thurstone, L. L., "The method of paired comparisons for social values.", The Journal of Abnormal and Social Psychology **21**, 4, 384 (1927). 1.2

Tideman, N., *Collective decisions and voting: the potential for public choice* (Routledge, 2017). 2.4

Truchon, M., *An extension of the Condorcet criterion and Kemeny orders* (Citeseer, 1998). 1.1, 2.4, 3.1.1, 11, 11

Van Zuylen, A. and D. P. Williamson, "Deterministic algorithms for rank aggregation and other ranking and clustering problems", in "International Workshop on Approximation and Online Algorithms", pp. 260–273 (Springer, 2007). 2.3.2

Werbin-Ofir, H., L. Dery and E. Shmueli, "Beyond majority: Label ranking ensembles based on voting rules", Expert Systems with Applications **136**, 50–61 (2019). 1.1

Xiao, Y., H.-Z. Deng, X. Lu and J. Wu, "Graph-based rank aggregation method for high-dimensional and partial rankings", Journal of the Operational Research Society **72**, 1, 227–236 (2021). 6

Xie, M., R. C.-W. Wong and A. Lall, "An experimental survey of regret minimization query and variants: bridging the best worlds between top-k query and skyline query", The VLDB Journal **29**, 1, 147–175 (2020). 1.1

Yoo, Y. and A. R. Escobedo, "A new binary programming formulation and social choice property for kemeny rank aggregation", Decision Analysis **18**, 4, 296–320 (2021). (document), 1.1, 1.2, 2, 2.1, 2.2, 2.3.1, 2.4, 3.2, 3.2.2, 3.1, 11, 3.3.4, 4.3.1, 22

Yoo, Y., A. R. Escobedo and J. K. Skolfield, "A new correlation coefficient for comparing and aggregating non-strict and incomplete rankings", European Journal of Operational Research (2020). 1.1, 2.3.1, 3.3.4, 3.3.4, 4.3.1, 6

Young, H. P., "Extending condorcet's rule", Journal of Economic Theory **16**, 2, 335–353 (1977). 2.4

Young, H. P., "Condorcet's theory of voting", American Political science review **82**, 4, 1231–1244 (1988). 1.1, 2.4

Young, H. P. and A. Levenglick, "A consistent extension of condorcet's election principle", SIAM Journal on applied Mathematics **35**, 2, 285–300 (1978). 1.1

Younger, D., "Minimum feedback arc sets for a directed graph", IEEE Transactions on Circuit Theory **10**, 2, 238–245 (1963). 3.3.2

Zhang, C., K. Cheng, L. Zhu, R. Chen, Z. Zhang and F. Huang, "Efficient continuous top-k geo-image search on road network", Multimedia Tools and Applications **78**, 21, 30809–30838 (2019). 1.1

Zhang, Y., T. Bouadi and A. Martin, "A clustering model for uncertain preferences based on belief functions", in "International Conference on Big Data Analytics and Knowledge Discovery", pp. 111–125 (Springer, 2018). 4.3

Zhang, Y., Y. Xiao, J. Wu and X. Lu, "Comprehensive world university ranking based on ranking aggregation", Computational Statistics **36**, 2, 1139–1152 (2021). 5.8

Zhu, Z., J. Wang and J. Caverlee, "Improving top-k recommendation via jointcollaborative autoencoders", in "The World Wide Web Conference", pp. 3483–3482 (2019). 1.1