mathematics

MDPI

Article

Reducing the Computational Time for the Kemeny Method by Exploiting Condorcet Properties

Noelia Rico [1,*], Camino R. Vela [1], Raúl Pérez-Fernández [2] and Irene Díaz [1]

[1] Department of Computer Science, University of Oviedo, 33203 Gijón, Spain; crvela@uniovi.es (C.R.V.); sirene@uniovi.es (I.D.)
[2] Department of Statistics and O.R. and Mathematics Didactics, University of Oviedo, 33007 Oviedo, Spain; perezfernandez@uniovi.es
* Correspondence: noeliarico@uniovi.es

Abstract: Preference aggregation and in particular ranking aggregation are mainly studied by the field of social choice theory but extensively applied in a variety of contexts. Among the most prominent methods for ranking aggregation, the Kemeny method has been proved to be the only one that satisfies some desirable properties such as neutrality, consistency and the Condorcet condition at the same time. Unfortunately, the problem of finding a Kemeny ranking is NP-hard, which prevents practitioners from using it in real-life problems. The state of the art of exact algorithms for the computation of the Kemeny ranking experienced a major boost last year with the presentation of an algorithm that provides searching time guarantee up to 13 alternatives. In this work, we propose an enhanced version of this algorithm based on pruning the search space when some Condorcet properties hold. This enhanced version greatly improves the performance in terms of runtime consumption.

Keywords: ranking aggregation; Condorcet; Kemeny method; exact algorithm; recursive method

check for updates

Citation: Rico, N.; Vela, C.R.; Pérez-Fernández, R.; Díaz, I. Reducing the Computational Time for the Kemeny Method by Exploiting Condorcet Properties. *Mathematics* **2021**, *9*, 1380. https://doi.org/10.3390/math9121380

Academic Editors: Santoso Wibowo and William Guo

Received: 13 May 2021
Accepted: 9 June 2021
Published: 15 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Elections in which several voters express their preferences over a set of alternatives in the form of rankings arise in many situations [1,2]. The problem of aggregating these rankings in order to rank the alternatives according to the preferences of the voters has been deeply studied in the field of social choice theory [3]. In particular, the problem can be traced back to the eighteenth century at the latest, when French scientist Condorcet [4] stated that whenever voters' preferences are expressed in the form of rankings, the only information that should be used is that given by the pairwise comparisons between the alternatives. According to this, Condorcet set a (simple) majority criterion principle based on choosing as winner of the election the alternative that beats all other alternatives by a majority of the votes, whenever such alternative exists.

When applying this majority criterion for establishing a winning ranking instead of a single winner, it is considered that an alternative should be ranked at a better position than another alternative in the winning ranking if the former defeats the latter by a majority of the votes. Unfortunately, the majority relation defined in this way is not necessarily transitive, and cycles of preferences might occur. This is referred to as the *voting paradox*. Even more so, the majority relation may be inconsistent with respect to any scoring method based on assigning points to the different alternatives according to their positions in the rankings, including the most prominent such method proposed by Borda [5].

Almost two centuries after Condorcet's proposal, Kemeny [6] proposed selecting the ranking that minimizes the distance to the rankings given by the voters [7]. This method can be understood as a natural extension of Condorcet's proposal since the ranking obtained by the Kemeny method coincides with the Condorcet ranking if it exists [8]. Unfortunately,

the computation of the Kemeny ranking for solving the Kemeny problem is known to be NP-hard [9].

The computation of ranking aggregation methods has gained attention in recent years, as they arise in real-life problems in several contexts [10–12]. The state-of-the-art exact algorithm for computing the solution(s) of the Kemeny method was introduced last year by Azzini and Munda [13]. In this paper, we review their proposal, pointing out a special case. We also propose a variation to ease the computation of the solutions based on Condorcet's criterion.

The remainder of this paper is organized as follows. Section 2 presents the Kemeny method. The state-of-the-art exact algorithm to resolve this problem is presented in Section 3. Sections 4 and 5 point out some modifications that can be incorporated into the algorithm. Three variations of the original algorithm are presented in Section 6. The experiments and the obtained results are presented and discussed in Sections 7 and 8. Final conclusions and future lines of research are outlined in Section 9.

2. The Kemeny Method

Consider a set of n alternatives $\mathcal{A} = \{a_1, \ldots, a_n\}$. Preferences over $\mathcal{A}$ are expressed in the form of a **ranking**, which is a complete, reflexive and transitive (but not necessarily antisymmetric) relation such that, for every pair of alternatives $a_i, a_j \in \mathcal{A}$, a strict order ($a_i \succ a_j$ or $a_j \succ a_i$) or equivalence ($a_i \sim a_j$) relation is defined. A strict order relation can be understood as a numeric vector that assigns an integer value in the interval $[1, n]$ to each alternative representing the position of the alternative in the ranking, being 1 the best possible position and consequently n the worst possible one.

The list of rankings given by the m different voters over the set of n alternatives is called the **profile of rankings**, denoted by π_m^n. As some voters may agree on their ranking, this would lead to a representation of the profile containing repeated rankings. In this work, we use the compact representation of the profile of rankings that only contains $m' \leq m$ unique rankings, where each ranking $r_i \in \pi_m^n$ is weighted by the number w_i of voters that expressed the ranking r_i. Thus, $m = \sum_{i=1}^{m'} w_i$.

The alternatives in $\mathcal{A}$ may be compared in a pairwise fashion by using a matrix $\mathbf{O}$ of dimension $n \times n$, known as the **outranking matrix** [14]. Each element o_{ij} of $\mathbf{O}$ represents the number of times that the alternative a_i is preferred over the alternative a_j by a voter. The value of the element o_{ij} is obtained from π_m^n by adding 1 point every time that $a_i \succ a_j$ holds in a ranking of the profile and 0.5 points every time that $a_i \sim a_j$ holds. Therefore, for each pair of alternatives a_i and a_j with $i \neq j$, it holds that $o_{ij} + o_{ji} = m$. By definition, all the elements of the diagonal are set to 0. This simpler representation gathers the most important information (pairwise information) provided by the profile of rankings.

An example of a profile of rankings and corresponding outranking matrix is shown in Table 1. In order to obtain the outranking matrix, the position of each pair of alternatives is evaluated in all the rankings of the profile. For example, to obtain the value of the element $o_{1,2}$ at the first row and second column, which represents the number of times that a_1 is preferred over a_2, the relative position of the alternatives a_1 and a_2 is checked over all the rankings. Every time that $a_1 \succ a_2$, meaning that the alternative a_1 is ranked at a better position than a_2, one point is added to $o_{1,2}$. For the first ranking of Table 1, it holds that $a_2 \succ a_1$, so no points are added to $o_{1,2}$. For the second ranking, it holds that $a_1 \succ a_2$; therefore, one point is added to $o_{1,2}$. For the third ranking, it also holds that $a_1 \succ a_2$; however, this ranking is expressed by four voters and therefore four points are added to $o_{1,2}$. For the fourth ranking, it holds that $a_2 \succ a_1$ and, thus, no point is added to $o_{1,2}$. In this way, it is therefore determined that $o_{1,2} = 5$. The element $o_{2,1}$ at the second row and first column may be calculated from $o_{1,2}$ due to the constant sum property of the matrix, as the number of voters is constant and therefore $o_{1,2}$ and $o_{2,1}$ must add up to the number of voters. In this example with 10 voters, $o_{2,1} = 10 - o_{1,2} = 5$.

Table 1. Profile of rankings π_{10}^4 given by ten voters on the set of four alternatives $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$ (left) and corresponding outranking matrix (right).

Number of Voters	Ranking
4	$a_3 \succ a_2 \succ a_4 \succ a_1$
1	$a_1 \succ a_2 \succ a_4 \succ a_3$
4	$a_4 \succ a_3 \succ a_1 \succ a_2$
1	$a_2 \succ a_1 \succ a_4 \succ a_3$

	a_1	a_2	a_3	a_4
a_1	0	5	2	2
a_2	5	0	2	6
a_3	8	8	0	4
a_4	8	4	6	0

The aim of a **ranking aggregation function** is to map the profile of rankings π_m^n into the ranking that best summarizes the information given by the voters, which is usually referred to as the *winning* or *consensus ranking*.

Condorcet stated that an alternative a_i should be ranked at a better position than another alternative a_j in the winning ranking if a_i is preferred by the majority of the voters over a_j, which in terms of the outranking matrix means that $o_{ij} > o_{ji}$. Unfortunately, as it was previously mentioned, this relation is not necessarily transitive as it may lead to a situation in which $a_i \succ a_j$, $a_j \succ a_k$ and $a_k \succ a_i$, even if the preferences were expressed in the form of complete rankings.

A prominent family of ranking aggregation functions is based on the use of a distance function δ on the set of rankings. The distance of a ranking to a profile of rankings $\delta(r, \pi_m^n)$ is computed by adding the individual distances from r to all rankings in π_m^n. From all the possible $n!$ complete rankings that can be obtained by permuting the set of n alternatives $\mathcal{A}$, the one (or ones) that minimizes the value of δ is selected as the winning ranking.

The most representative example of this family of distance-based methods is the one proposed by Kemeny [6]. According to Kemeny, the distance between two rankings is the number of discrepancies in the relative order of every pair of alternatives. Formally, two points are added every time that two alternatives appear in the rankings in the opposite order and 1 point is added every time that two alternatives are tied in exactly one of the rankings. Thus, the distance of a ranking s to the profile of rankings π_m^n is defined as the sum of the Kemeny distances from s to all the rankings $r_i \in \pi_m^n$.

The Kemeny ranking has been referred to in the literature as the **maximum likelihood ranking**. This is due to an equivalent interpretation where, instead of seeking for the ranking that minimizes the distance to the profile of rankings, we seek for the ranking that maximizes the agreement with the profile of rankings. The Kemeny ranking(s) can be computed from the outranking matrix by searching for the ranking(s) with maximum agreement score σ, defined as

$$\sigma(s, \pi_m^n) = \sum_{i=1}^{n} \sum_{j=1}^{n} o_{ij} \cdot x_{ij}, \tag{1}$$

where $x_{ij} = 1$ if $a_i \succ^s a_j$ and 0 otherwise.

For the profile of rankings in Table 1, the agreement scores for all possible rankings on $\mathcal{A}$ are shown in Table 2. It can be seen that the agreement score is maximized by the rankings $a_3 \succ a_2 \succ a_4 \succ a_1$, $a_4 \succ a_3 \succ a_1 \succ a_2$ and $a_4 \succ a_3 \succ a_2 \succ a_1$. These three rankings are the solutions to the Kemeny problem for the profile of rankings in Table 1.

Young [8] observed that the Kemeny method returns the Condorcet ranking in case it exists. Furthermore, it is known that the Kemeny method is the only ranking aggregation method that is neutral, consistent, and Condorcet [15,16]. Unfortunately, the problem of finding a Kemeny ranking has been proved to be NP-hard [9], which prevents its use in practice in many real-life problems. There exists no known algorithm to compute the Kemeny ranking in polynomial time, for any given number of alternatives greater than or equal to 3.

Table 2. Agreement score σ for all the possible rankings according to the Kemeny method for the profile of rankings in Table 1.

Ranking	σ	Ranking	σ	Ranking	σ	Ranking	σ
$a_3 \succ a_2 \succ a_4 \succ a_1$	**39**	$a_2 \succ a_3 \succ a_4 \succ a_1$	33	$a_2 \succ a_4 \succ a_1 \succ a_3$	29	$a_1 \succ a_3 \succ a_4 \succ a_2$	25
$a_4 \succ a_3 \succ a_1 \succ a_2$	**39**	$a_3 \succ a_2 \succ a_1 \succ a_4$	33	$a_1 \succ a_3 \succ a_2 \succ a_4$	27	$a_1 \succ a_2 \succ a_4 \succ a_3$	23
$a_4 \succ a_3 \succ a_2 \succ a_1$	**39**	$a_3 \succ a_1 \succ a_2 \succ a_4$	33	$a_1 \succ a_4 \succ a_3 \succ a_2$	27	$a_2 \succ a_1 \succ a_4 \succ a_3$	23
$a_3 \succ a_4 \succ a_1 \succ a_2$	37	$a_4 \succ a_1 \succ a_3 \succ a_2$	33	$a_2 \succ a_3 \succ a_1 \succ a_4$	27	$a_1 \succ a_2 \succ a_3 \succ a_4$	21
$a_3 \succ a_4 \succ a_2 \succ a_1$	37	$a_4 \succ a_2 \succ a_3 \succ a_1$	33	$a_4 \succ a_1 \succ a_2 \succ a_3$	27	$a_1 \succ a_4 \succ a_2 \succ a_3$	21
$a_2 \succ a_4 \succ a_3 \succ a_1$	35	$a_3 \succ a_1 \succ a_4 \succ a_2$	31	$a_4 \succ a_2 \succ a_1 \succ a_3$	27	$a_2 \succ a_1 \succ a_3 \succ a_4$	21

3. Azzini and Munda's Algorithm

The state-of-the-art exact algorithm for the computation of the Kemeny ranking(s) was presented by Azzini and Munda in [13] and is based on the theoretical equivalence between the Kemeny ranking and the maximum likelihood ranking. The algorithm considers a recursive process based on outranking matrices to compute σ, which makes the computational time only depend on the number of alternatives to be considered, so it does not increase as the number of voters increases. The obtained results improved the computational time by the exact algorithm used as benchmark up to that moment. Azzini and Munda's algorithm is based on the two following propositions:

Proposition 1 ([13]). *Let $\mathcal{A}$ be a set of n alternatives and let $\mathbf{O} = [o_{ij}]$ be an outranking matrix defined on $\mathcal{A}$. There exists at least one alternative $a_i \in \mathcal{A}$ such that $\sum_{j=1}^{n} o_{ij} \geq \sum_{j=1}^{n} o_{ji}$.*

Proposition 2 ([13]). *Let $\mathcal{A}$ be a set of n alternatives and let $\mathbf{O} = [o_{ij}]$ be an outranking matrix defined on $\mathcal{A}$. A necessary condition for a ranking to have the maximum likelihood score is that the alternative at the top position $a_i \in \mathcal{A}$ satisfies that $\sum_{j=1}^{n} o_{ij} \geq \sum_{j=1}^{n} o_{ji}$.*

Note that not all the alternatives whose row sum is greater than or equal to their corresponding column sum are ranked at the first position in one of the solutions of the Kemeny problem.

For the sake of simplicity, in the remainder of this paper we denote by α_i the truth value obtained from the Boolean expression:

$$\alpha_i = \left(\sum_{j=1}^{n} o_{ij} \geq \sum_{j=1}^{n} o_{ji} \right), \tag{2}$$

meaning that α_i is *True* when the alternative a_i is preferred over the remaining alternatives in $\mathcal{A}$ at least as many times as the other alternatives are preferred over a_i.

From Proposition 2 it is deduced that the alternative a_i at the first position of the ranking must always be one such that $\alpha_i = True$. From a computational point of view, this allows for a great reduction in the set of rankings to explore as possible solutions.

The starting point of our work is the **Mork-Exact** algorithm (consequence of Proposition 2), presented in [13], which is outlined below:

Step p. In the current outranking matrix with ($n \geq 3$), choose all alternatives whose row sum is not smaller than the corresponding column sum.

Step p+1. Choose one of the alternatives found in *Step p* and delete its corresponding row and column.

Step p+2. Given the new current outranking matrix, repeat *Step p* and *Step p+1*. The algorithm stops when the outranking matrix becomes a matrix of dimension 2×2, in this case choose the alternative with largest concordance index (i.e., the largest nondiagonal value). Steps *p+1* and *p+2* are repeated for each alternative whose row sum is bigger than or equal to its column sum found in all steps.

Final step. Compute the total score for all the rankings obtained in the previous steps. Delete all rankings whose total score is not the maximum one. Note that this step is needed since Proposition 2 gives a necessary but not sufficient condition.

In this paper, we propose an improvement for the Mork-Exact algorithm with the aim of decreasing its execution time and make it more efficient in computational terms.

4. Constraints Based on the Condorcet Conditions

4.1. The Condorcet Ranking

The *Condorcet ranking* is a ranking such that each alternative is preferred by more than half of the number of the voters to all other alternatives ranked at a worse position. The Condorcet ranking might not exist for a given profile of rankings; however, it is unique in case of existence. The Kemeny method returns the Condorcet ranking in case it exists for the profile of rankings [15].

In case of existence of the Condorcet ranking, the outranking matrix must be transitive. In such case, the algorithm could avoid the execution of the recursive process, since the winning ranking can be determined faster from the outranking matrix by counting how many times each alternative $a_i \in \mathcal{A}$ is preferred over other alternatives.

Let h be the value representing half of the number of voters, more precisely, $h = \frac{m}{2}$ if m is even and $h = \frac{m-1}{2}$ if it is odd. The Boolean outranking matrix $\mathbf{B}$ is obtained from the outranking matrix $\mathbf{O}$ such that $b_{ij} = 1$ if $o_{ij} > h$ and $b_{ij} = 0$ otherwise. Let us denote the sum of the elements at the i-th row of the Boolean matrix as $\beta_i = \sum_{j=1}^{n} b_{ij}$.

When there exists a Condorcet ranking, the sum of the elements of each row in $\mathbf{B}$ gives for each alternative a different integer value in the interval $[0, n-1]$. Thus, the Condorcet ranking is obtained by sorting these values in descending order.

This can be used as a precondition in the algorithm in order to determine whether the recursive process used to find the Kemeny ranking may be skipped.

4.2. The Condorcet Winner

The *Condorcet winner* is an alternative that is preferred by more than half of the number of the voters to all other alternatives. The Kemeny method is known to be a *Condorcet method*, which means that, in case the Condorcet winner exists, this alternative is ranked at the first position.

It is known that in the cases in which the profile of rankings does not yield a Condorcet ranking, $\mathbf{B}$ is not transitive. If there exists a Condorcet winner, then the row corresponding to this alternative in $\mathbf{O}$ contains $n-1$ values greater than h, as the alternative a_i is preferred over all other alternatives by more than half of the number of voters. This means that the column associated with this alternative has a value $\beta = n-1$. Therefore, for the outranking matrices where this value of β is present but the Condorcet ranking does not exist, the Condorcet winner still exists. Notice that only one alternative may be the Condorcet winner.

However, although there exists at most one Condorcet winner, there could be more than one alternative such that $\alpha_i = \mathit{True}$. If one of these alternatives is the Condorcet winner, the exploration of all other elements with $\alpha_i = \mathit{True}$ could be omitted, which may lead to a significant decrease in the execution time of the algorithm.

5. Tied Alternatives in the Recursive Process

Step p+2 of Mork-Exact states that '*the algorithm stops when the outranking matrix becomes a matrix of dimension* 2×2*, in this case choose the alternative with the largest concordance index*'. At this point, this matrix of dimension 2×2 is of the following form:

$$\begin{pmatrix} 0 & x \\ m-x & 0 \end{pmatrix},$$

for the two remaining alternatives a_i and a_j, where x is the number of times that a_i is preferred over a_j due to the property of constant sum $o_{ij} + o_{ji} = m$ of the outranking matrix.

Notice that there is a special case that can be found in some profiles of rankings. Consider a profile of rankings with an even number of voters m. If two alternatives $a_i, a_j \in \mathcal{A}$ that are tied (each of them is preferred to the other one by $h = \frac{m}{2}$ voters) are considered at the end of the recursive process, then the matrix of dimension 2×2 obtained from the recursive process has two equal concordance indexes, i.e., $x = m - x = h$. In this case Mork-Exact does not determine what to do as the *largest* concordance index does not exist. It is necessary for the implementation of the algorithm to take into account this scenario, which implies that the recursive process must return both rankings as tentative solutions.

A profile of rankings for which this situation occurs is presented in Table 1. The execution trace for the algorithm is illustrated in Figure 1.

Step 1:

	a_1	a_2	a_3	a_4	
a_1	0	5	2	2	$\longrightarrow$ 9 ✗
a_2	5	0	2	6	$\longrightarrow$ 13 ✗
a_3	8	8	0	4	$\longrightarrow$ 20 ✓
a_4	8	4	6	0	$\longrightarrow$ 18 ✓

Two recursive calls:
$a_3 \succ \ldots$
$a_4 \succ \ldots$

Step 2: $a_3 \succ \ldots$

	a_1	a_2	a_4	
a_1	0	5	2	$\longrightarrow$ 7 ✗
a_2	5	0	6	$\longrightarrow$ 11 ✓
a_4	8	4	0	$\longrightarrow$ 12 ✓

Two recursive calls:
$a_3 \succ a_2 \ldots$
$a_3 \succ a_4 \ldots$

Step 3: $a_3 \succ a_2 \ldots$

	a_1	a_4	
a_1	0	2	$\longrightarrow$ 2 ✗
a_4	8	0	$\longrightarrow$ 8 ✓

Winning alternative a_4, return:
$a_3 \succ a_2 \succ a_4 \succ a_1$

Step 4: $a_3 \succ a_4 \ldots$

	a_1	a_2	
a_1	0	5	$\longrightarrow$ 5 ✓
a_2	5	0	$\longrightarrow$ 5 ✓

Tie, return both rankings:
$a_3 \succ a_4 \succ a_1 \succ a_2$
$a_3 \succ a_4 \succ a_2 \succ a_1$

Step 5: $a_4 \succ \ldots$

	a_1	a_2	a_3	
a_1	0	5	2	$\longrightarrow$ 7 ✗
a_2	5	0	2	$\longrightarrow$ 7 ✗
a_3	8	8	0	$\longrightarrow$ 16 ✓

One recursive call:
$a_4 \succ a_3 \succ \ldots$

Figure 1. *Cont.*

Step 6: $a_4 \succ a_3 \ldots$

	a_1	a_2		
a_1	0	5	$\longrightarrow$	5 ✓
a_2	5	0	$\longrightarrow$	5 ✓

Tie, return both rankings:

$a_4 \succ a_3 \succ a_1 \succ a_2$

$a_4 \succ a_3 \succ a_2 \succ a_1$

Figure 1. Recursive process followed by the algorithm over the profile of rankings in Table 1 in order to determine the tentative solutions, whose score will be later computed to determine the winning ranking(s) according to the Kemeny method.

After the recursive process is terminated, the agreement score of the rankings in the list of tentative solutions for the profile of rankings must be computed. The obtained agreement scores are the following ones:

- $a_3 \succ a_2 \succ a_4 \succ a_1$ ($\sigma = 39$)
- $a_3 \succ a_4 \succ a_1 \succ a_2$ ($\sigma = 37$)
- $a_3 \succ a_4 \succ a_2 \succ a_1$ ($\sigma = 37$)
- $a_4 \succ a_3 \succ a_1 \succ a_2$ ($\sigma = 39$)
- $a_4 \succ a_3 \succ a_2 \succ a_1$ ($\sigma = 39$)

From this list, only the ones maximizing the agreement score (in other words, the ones that are the closest to the profile of rankings) are kept. These rankings are $a_3 \succ a_2 \succ a_4 \succ a_1$, $a_4 \succ a_3 \succ a_1 \succ a_2$ and $a_4 \succ a_3 \succ a_2 \succ a_1$. Thus, the Kemeny method applied to this profile of rankings admits three possible solutions.

Therefore, we conclude that it is necessary to include these rankings in the list of rankings to explore as these rankings might be Kemeny rankings. Otherwise, the obtained list of Kemeny rankings might be incomplete.

6. Proposed Algorithms

In this work, we propose three variations of the Mork-Exact algorithm based on the considerations introduced in the previous sections:

- `ME`: Implementation of the original algorithm taking into account the case in which more than one ranking must be returned in the recursive process. It also incorporates the precondition that allows one to skip the recursive process in first place in case the Condorcet ranking exists for the given profile of rankings.
- `ME-CW`: Checks whether the Condorcet winner exists for the given profile of rankings. If this is the case, then the recursive call of the first level is made only for the Condorcet winner and not for all the other alternatives with a value of $\alpha_i = True$.
- `ME-RCW`: If the Condorcet winner is found in the recursive call, then the ranking of the remaining alternatives that minimizes the distance to the profile in the recursive call must start with this alternative. This algorithm recursively checks the existence of the Condorcet winner and in case the Condorcet winner exists at any level it avoids the exploration of the alternatives that have a value of $\alpha_i = True$ but are not the Condorcet winner (if any).

The results are presented for these three algorithms. Please note that the algorithm that checks whether the Condorcet ranking exists for the remaining alternatives in the recursive process has also been tested. In this case, the results are not improved for all the profiles of rankings. The reason is that, for those profiles for which the Condorcet ranking does not exist at any point, the cost of recursively computing if the Condorcet ranking exists increases the execution time without providing any advantage. Therefore, this algorithm has been omitted in the upcoming sections.

As it is not possible to know in advance the number of recursive calls, the study of the complexity is not trivial. Furthermore, the problem is still NP-hard as the proposed algorithms cannot guarantee that the obtained complexity could be polynomial. In relation to the execution time, this depends on the number of alternatives that fulfill the condition

$\alpha = True$, which determines the number of recursive calls. In a more general sense, the execution time of these algorithms depends on the number of rankings that are potential solutions (which is at most $n!$) and how many of them can be discarded by taking into account Proposition 2. The proposed algorithms also incorporate the Condorcet conditions to discard potential solutions and thus reduce the execution time. For example, the best situation for the `ME-CW` algorithm occurs when there exists a Condorcet winner and therefore only $\frac{n!}{n} = (n-1)!$ rankings are potential solutions to the Kemeny problem in the first recursive call (and some of them are discarded in the next steps). The `ME-RCW` algorithm follows a similar approach at different levels of the execution, which greatly reduces the number of tentative solutions and, consequently, the execution time. Notice how the execution time is mainly dominated by the number of alternatives, as the size of the input data of the algorithm is always a matrix of dimension $n \times n$, no matter the number of voters.

7. Experiments

In the original paper in which the Mork-Exact algorithm was proposed [13], the authors provide the average execution time of the algorithm for different numbers of alternatives. To measure this execution time, they randomly generated 1000 different matrices for each $n \in [3, 13]$, with a fixed number of voters equal to 100.

However, as it is highlighted in [17], the execution time of the algorithms does not depend only on the number of alternatives in the profile of rankings but also in the difficulty of the profile of rankings itself. As the computation time of the algorithms considered in this work strongly relies on the number of alternatives with $\alpha = True$, we have defined the difficulty of the profiles according to the number of alternatives satisfying $\alpha = True$. Let ω be the number of alternatives in the profile satisfying $\alpha = True$, formally defined as

$$\omega = \sum_{a_i \in \mathcal{A}} \alpha_i \,. \tag{3}$$

The value of ω in a profile of rankings with n alternatives ranges in the interval of natural numbers $[1, n]$. Note that ω depends on the votes distribution.

The value of ω has a high impact on the execution time of the algorithm, as it determines the number of recursive calls that are made at the first level of the matrix. This has a strong relation with the number of rankings considered as tentative solutions and therefore with the impact of the time required to solve this problem.

For our experiments, we have designed two different lists of profiles of rankings in order to test the algorithm under different conditions: (1) a list containing profiles of rankings for which the Condorcet winner exists but the Condorcet ranking does not exist and (2) a list with profiles of rankings for which the Condorcet winner does not exist (and consequently the Condorcet ranking does not exist either). Each list contains for each pair (n, ω) with $n \in [8, 12]$ (we focus on these values of n because the solution for $n < 8$ when using this algorithm is instantly returned) and $\omega \in [1, n-1]$, 5 different profiles of rankings that have been synthetically generated with a constant number of voters. With this experimental setting it is ensured that profiles associated with all ranges of execution time are considered in the evaluation of the algorithm. This allows to explore the worst-case execution time of the algorithm, presented when all the alternatives are such that $\alpha = True$, even if that scenario is not common in real-life datasets. Furthermore, we have also created two shorter lists of 30 and 25 profiles of rankings of 13 and 14 alternatives, respectively, for which the Condorcet winner does not exist. The generated profiles of rankings are available at https://github.com/noeliarico/consensus_benchmark (accessed on 12 May 2021).

Considering the lack of access to the original code and experiments and the fact that the algorithms have been tested by using different processors (original results are obtained by using an Intel Core i7, whereas the here-presented results are obtained by using an Apple M1), a fair comparison with the algorithm proposed in [13] is not possible. For

this very reason, the execution time of the algorithm ME will be used as a baseline for measuring the improvement attained by the new versions of the algorithm. This ME version is the most efficient version of the algorithm that we have obtained after considering different approaches and making use of different data structures and memory optimization techniques in Python 3.7. Therefore, the results presented in the next section are given based on the relative reduction of the execution time in relation to ME.

The aim of these experiments is to analyze the behavior of the three proposed algorithms for every profile of rankings in the generated lists. In order to minimize the impact of other processes being executed by the computer during the measurements when the execution times are low (i.e., up to $n = 11$), each experiment has been repeated three times and the median value of these executions (to avoid the effect of possible outliers if computing the mean) has been taken into account. For each pair of values (n, ω), the average time obtained from executing the algorithms for the five profiles of rankings with similar characteristics has been considered.

8. Results

The results of the above-detailed experiments are presented in this section. Figure 2 presents the results obtained for the list of profiles of rankings for which the Condorcet winner exists. Each bar represents the average time for a different number of alternatives, showing that the execution time of the basic algorithm ME is reduced by at least 75% for the ME-CW algorithm and by around 90% for the ME-RCW. Note that the Condorcet winner exists for 28–77% (depending on the values of n and m) of the generated profiles of rankings.

The results for profiles of rankings for which the Condorcet winner does not exist are presented in Figure 3, where these are compared in terms of execution time with the original ME algorithm, showing improved times for any number of alternatives. In this case, the ME-RCW also reduces the execution time, reaching a reduction of almost 50% of the execution time when the number of alternatives increases. The results of the ME-CW are not presented in this plot, as this algorithm only reduces the execution time when the Condorcet winner exists. Nevertheless, it is important to remark that the execution time does not increase for profiles of rankings when the Condorcet winner does not exist, as the operation to check the existence of the Condorcet winner can be executed in less than 1ms and it is only performed once at the beginning of the algorithm.

For a better understanding of the results, a detailed visualization is presented in Figure 4. The x-axis in both plots of the figure has been obtained with a $\log_{10}$ transformation of the number of tentative solutions explored for each profile. On the left-hand side, a complete view of the results given by the algorithms ME and ME-RCW for the profiles of rankings with $n = 12$ for which the Condorcet winner does not exist is shown. Notice that the execution time soars for high values of ω in the basic implementation of the algorithm ME according to our experiments. On the right-hand side, the same plot is zoomed highlighting the values of $\omega < 8$, where a similar behavior can be observed. Note that the greater the value of ω is, the bigger is the gap in the execution time between both algorithms.

Due to the increase of the execution time of ME in extreme situations, we have measured the execution time for $n = 13$ and $n = 14$ only for the improved algorithm ME-RCW and avoided large values of ω. The results are presented in Table 3. Note that the execution time varies widely depending on the value of ω, but the algorithm is still executed in reasonable time and it does not show any memory requirement problem. For profiles of rankings with a small value of ω, the number of recursive calls made at the first level of the algorithm is reduced, allowing many tentative solutions from the very beginning to be discarded. Therefore, aside from large values of ω, the execution time is affordable even for this number of alternatives.

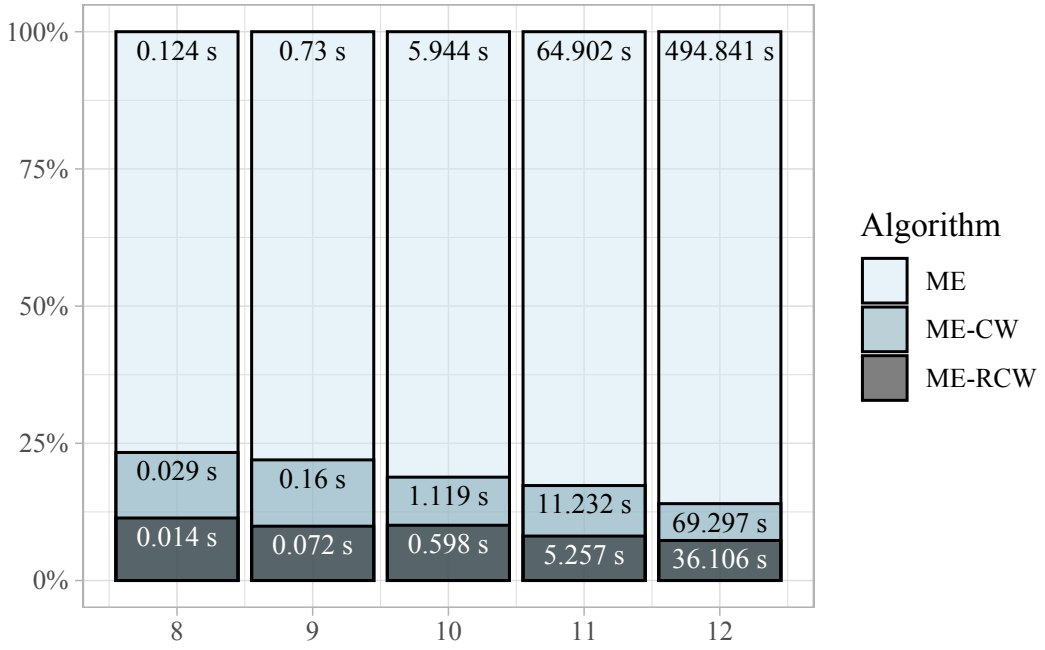

Figure 2. Execution time in seconds and percentage of time reduction of the three algorithms for the profiles of rankings for which the Condorcet winner exists. Results are presented for different numbers of alternatives.

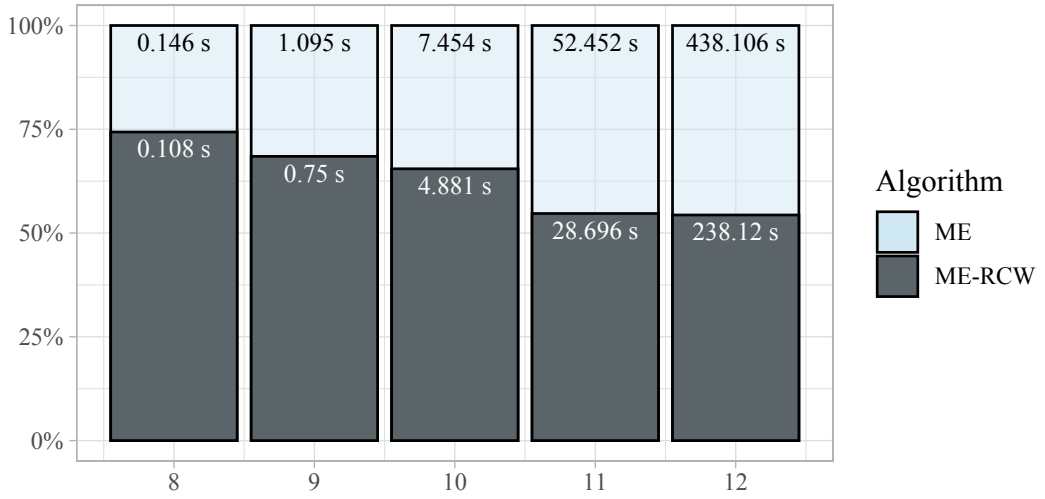

Figure 3. Average execution time in seconds and percentage of time reduction of the `ME-RCW` algorithm for the profiles of rankings for which the Condorcet winner does not exist. Results are presented for different numbers of alternatives.

Table 3. Execution time for profiles of rankings of 13 and 14 alternatives for which the Condorcet winner does not exist.

	ω		Time (in Seconds)			
n	min	max	min	Median	Mean	max
13	3	8	6.52	105.93	278.25	1935.66
14	3	7	159.90	579.86	900.38	4002.10

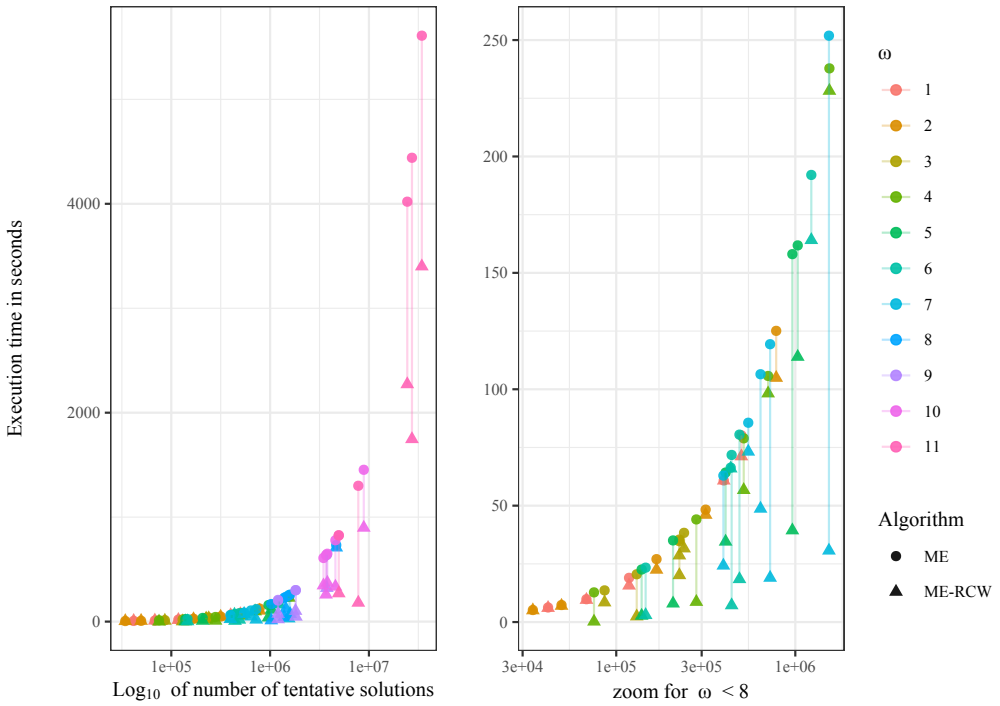

Figure 4. Reduction of the execution time with `ME-RCW` for profiles of rankings of $n = 12$ that do not have a Condorcet winner. Each plot shows the execution time in the y-axis and the $\log_{10}$ transformation of the number of tentative solutions explored in the x-axis. Each line represents, for one profile of rankings, the reduction time of the algorithm `ME-RCW` in relation to `ME`. Lines are colored according to the value of ω of the profile.

The obtained results show that the proposed algorithms for the computation of the Kemeny rankign(s) outperform the state-of-the-art exact algorithm [13], which already outperformed greatly other exact algorithms proposed in the literature [18,19].

9. Conclusions

In this work, we proposed some variations of the exact algorithm recently proposed by Azzini and Munda for the computation of the Kemeny ranking(s). These variations are proved to improve the efficiency of the algorithm. Furthermore, we have stressed the influence of the characteristics of the profile of rankings on the execution time of the algorithms. Future work will consider a more detailed revision of the characteristics of the profiles of rankings and their influence on the performance as well as the implementation of more elaborate techniques for reducing the exploration of tentative solutions in order to find the Kemeny ranking.

Author Contributions: Conceptualization, N.R., I.D.; Formal analysis, C.R.V.; Funding acquisition, I.D.; Investigation, N.R.; Methodology, N.R. and R.P.-F.; Software, N.R.; Validation, C.R.V., R.P.-F. and I.D.; Writing—original draft, N.R.; Writing, review, editing, C.R.V., R.P.-F. and I.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by Grant TIN2017-87600-P from the Spanish Government and PID2019-106263RB-I00. Noelia Rico is supported by the Severo Ochoa program (PA-20-PF-BP19-167).

Data Availability Statement: https://github.com/noeliarico/consensus_benchmark.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goerlich, F.J.; Reig, E. Quality of life ranking of Spanish cities: A non-compensatory approach. *Cities* **2021**, *109*, 102979. [CrossRef]
2. Bianchi, A.; Biffignandi, S. Workplace Social Environment Indicator: A Comparative Analysis of European Regions. *Soc. Indic. Res.* **2020**, 1–20. [CrossRef]
3. Gaertner, W. *A Primer in Social Choice Theory: Revised Edition*; Oxford University Press: Oxford, UK, 2009.
4. Condorcet, M. *Essai sur l'Application de l'Analyse à la Probabilité des Décisions Rendues à la Pluralité des Voix*; De l'Imprimerie Royale: Paris, France, 1785.
5. Borda, J.C. *Mémoire sur les Élections au Scrutin*; Histoire de l'Académie Royale des Sciences: Paris, France, 1781.
6. Kemeny, J.G. Mathematics without Numbers. *Daedalus* **1959**, *88*, 577–591.
7. Kemeny, J.G.; Snell, J.L. *Mathematical Models in the Social Sciences*; The MIT Press: Cambridge, MA, USA, 1972.
8. Young, H.P. Condorcet's theory of voting. *Am. Political Sci. Rev.* **1988**, *82*, 1231–1244. [CrossRef]
9. Bartholdi, J.; Tovey, C.A.; Trick, M.A. Voting Schemes for which It Can Be Difficult to Tell Who Won the Election. *Soc. Choice Welf.* **1989**, *6*, 157–165. [CrossRef]
10. Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; Procaccia, A.D. (Eds.) *Handbook of Computational Social Choice*; Cambridge University Press, Cambridge, UK, 2016.
11. Pérez-Fernández, R.; Rademaker, M.; Alonso, P.; Díaz, I.; Montes, S.; De Baets, B. Representations of votes facilitating monotonicity-based ranking rules: From votrix to votex. *Int. J. Approx. Reason.* **2016**, *73*, 87–107. [CrossRef]
12. Pérez-Fernández, R.; Alonso, P.; Díaz, I.; Montes, S.; De Baets, B. Monotonicity as a tool for differentiating between truth and optimality in the aggregation of rankings. *J. Math. Psychol.* **2017**, *77*, 1–9. [CrossRef]
13. Azzini, I.; Munda, G. A new approach for identifying the Kemeny median ranking. *Eur. J. Oper. Res.* **2020**, *281*, 388–401. [CrossRef]
14. Arrow, K.; Raynaud, H. *Social Choice and Multicriterion Decision-Making*, 1st ed.; The MIT Press, Cambridge, MA, USA, 1986; Volume 1.
15. Young, H.P.; Levenglick, A. A Consistent Extension of Condorcet's Election Principle. *SIAM J. Appl. Math.* **1978**, *35*, 285–300. [CrossRef]
16. Hemaspaandra, E.; Spakowski, H.; Vogel, J. The complexity of Kemeny elections. *Theor. Comput. Sci.* **2005**, *349*, 382–391. [CrossRef]
17. Ali, A.; Meilă, M. Experiments with Kemeny ranking: What works when? Computational Foundations of Social Choice. *Math. Soc. Sci.* **2012**, *64*, 28–40. [CrossRef]
18. Muravyov, S.V. Ordinal measurement, preference aggregation and interlaboratory comparisons. *Measurement* **2013**, *46*, 2927–2935. [CrossRef]
19. Amodio, S.; D'Ambrosio, A.; Siciliano, R. Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. *Eur. J. Oper. Res.* **2016**, *249*, 667–676. [CrossRef]