Aggregate your rankings with CoRankCo

Pierre Andrieu
LRI, CNRS, U. Paris-Sud
U. Paris-Saclay, France
pierre.andrieu@lri.fr

Bryan Brancotte
IFB-Core
CNRS, INRIA, INRA
France
bryan.brancotte@france-bioinformatique.fr

Sarah Cohen-Boulakia
LRI, CNRS, U. Paris-Sud
U. Paris-Saclay, France
cohen@lri.fr

Alain Denise
LRI, CNRS, U. Paris-Sud
I2BC, CNRS, CEA
U. Paris-Sud
U. Paris-Saclay, France
alain.denise@u-psud.fr

Robin Milosz
DIRO, U. Montreal
Quebec, Canada
miloszro@iro.umontreal.ca

Adeline Pierrot
LRI, CNRS, U. Paris-Sud
U. Paris-Saclay, France
adeline.pierrot@lri.fr

ABSTRACT

Aggregating multiple rankings into one consensus ranking is a need in a very large number of contexts including aggregating answers returned by several web engines, computing a global rating based on numerous user ratings, determining the winner in a sport competition. Rank aggregation (or consensus ranking) is an active research topic, especially in the database community. A plethora of approaches have been developed to compute one consensus ranking from a set of input rankings. However, most approaches have considered data sets where input rankings are permutations: each ranking contains the exact same set of elements and elements are strictly ordered. In real applications, the rankings to be aggregated may not be permutations as they may be incomplete (input rankings may not consider all the same elements) and may have ties (where some elements are placed at the same position, considered as *ex aequo*). New measures must then be considered and existing approaches must be adapted to this new context when possible.

In this demo paper, we introduce CoRankCo, the first platform based on a generic consensus ranking framework, able to compute consensus rankings online while considering incomplete rankings and rankings with ties. With CoRankCo, users can (re)use various data sets to run a large variety of consensus ranking algorithms, possibly parameterized by various measures.

PVLDB Reference Format:
Pierre Andrieu, Bryan Brancotte, Sarah Cohen-Boulakia, Alain Denise, Robin Milosz, Adeline Pierrot, Stéphane Vialette. Aggregate your rankings with CoRankCo. *PVLDB*, 11 (5): xxxx-yyyy, 2018.
DOI: https://doi.org/TBD

Proceedings of the VLDB Endowment, Vol. 11, No. 5

DOI: https://doi.org/TBD

1. MOTIVATION

The problem of aggregating multiple rankings into one consensus ranking has started to be investigated two centuries ago and has been actively studied again in the last decades. Direct applications are numerous and include aggregating answers returned by several web engines [7], computing a global rating based on numerous user ratings [7] or determining the winner in a sport competition [4]. This topic has been of particular interest in the information retrieval and database communities ([7] and [8, 9]) while several other communities have also deeply looked into it, from algorithmics ([1]) to social sciences ([2]).

As the intrinsic problem is NP-hard, a plethora of approaches have been developed to compute one consensus ranking from a set of input rankings (providing exact algorithms, approximations, and heuristics). Most approaches have considered data sets where input rankings are permutations: each ranking contains the exact same set of elements and elements are strictly ordered. However, in real applications, the rankings to be aggregated may not be permutations as they may be incomplete (input rankings may not consider all the same elements) and may have ties (where some elements are placed at the same position, considered as *ex aequo*). Measures used to determine how rankings differ must then be (re)designed to take into account such new settings. Existing approaches must also be adapted to consider these new measures.

In this paper, we claim that when computing consensus rankings of real datasets, the choice of the measure used to determine how rankings differ must actually be guided by the interpretation to be given to the missing elements.

To illustrate this point, we now provide two use cases, both based on the same set of rankings provided below. The challenge lies in providing a consensus of possibly incomplete rankings able to both highlight the common points between input rankings and minimize their disagreements.

```
r1 := [ {A}, {B}, {C}, {D}, {E} ]
r2 := [ {B}, {A}, {G}, {C}, {F}, {E} ]
r3 := [ {A}, {G}, {C}, {D}, {B} ]
r4 := [ {H}, {A}, {C}, {B}, {G}, {D}, {F}, {E} ]
r5 := [ {H}, {B}, {A}, {D}, {F}, {C}, {G}, {E} ]
```

Use case 1 (Web crawlers). Consider a first use case where a set of five crawlers (r1,...,r5) have provided rankings searching for the most important web pages to be consulted. Depending on the web crawler, five to eight web pages (A to G) have been ranked.

How should missing elements be interpreted and managed? Here, web pages which have not been returned by a given crawler are equally irrelevant for that crawler. In other words, missing elements for a given ranking can be artificially placed into a *unifying bucket* at the last position of such a ranking. For instance, r_1 can be modified by adding the following bucket {F, G, H} at its last position (6^{th}).

What would be the expected outcome? Consider arbitrarily two elements: A and H. Intuitively, the page A should be ranked high in the consensus ranking since it is ranked twice at the first position (in r1 and r3), twice at the second position (in r2 and r4) and once at the third position (in r5). On the other hand, H is a less good candidate than A as it is ranked twice at the first position but not even in the top-five of the other rankings. In other words, in this setting, it is expected that the consensus ranking places $\underline{A \text{ before } H}$.

Use case 2 (Movie fans). Continuing with the same dataset, let us now consider a totally different context: elements are movies and rankings are provided by fans who have ranked the top movies they have seen.

How should missing elements be interpreted and managed? Here, a missing element in a given ranking corresponds to a movie that a given fan has not seen. Contrarily to the previous use case, it now does not make sense to compare seen and unseen movies: missing and present elements cannot be compared to each others'.

What would be the expected outcome? Consider again carefully the set of rankings and focus again on elements A and H. Note now that the fan who have watched both movie H and A, have systematically preferred H over A. So, contrarily to the previous case, it is expected that a consensus ranking places $\underline{H \text{ before } A}$.

Depending on the context, missing elements must thus be carefully interpreted for the right measure to be used when computing the consensus ranking.

In this demo paper, we introduce CoRankCo, the first platform based on a generic consensus ranking framework, able to compute consensus rankings online while considering incomplete rankings and rankings with ties. With CoRankCo, users can (re)use various data sets to run a large variety of consensus ranking algorithms, parameterized by various measures. In the rest of the paper we introduce the formal framework in which rankings and measures can be defined (Section 2), present the main functionalities of the CoRankCo platform (Section 3) and provide the sketch of the demonstration (Section 4).

2. RANK AGGREGATION

There are two main kinds of rank aggregation approaches [6]: *score-based* approaches which are based on the search for a consensus minimizing a score which depends on a distance measure between rankings, and *positional* approaches which use the position of the elements in the rankings to compute a consensus. This section is devoted to *score-based* approaches as they have been the most studied in the last decade.

2.1 Classical framework

The rank aggregation problem has been originally defined for a set of permutations. A **permutation** π is a bijection of $[n] = \{1, 2 \ldots, n\}$ onto itself. It represents a strict total order of the elements of $[n]$ (it is thus a *ranking*). As usual, $\pi[i]$ stands for the image (or position) of integer i in permutation π, and we denote $\pi = \pi[1]\pi[2] \ldots \pi[n]$.

A classical dissimilarity measure for comparing two permutations is the **Kendall-τ distance** which counts the number of pairs for which the order is different in the two permutations. Formally, the Kendall-τ distance, here denoted D, is defined as:

$$D(\pi, \sigma) = |\{(i,j) : i < j \ \wedge \ (\pi[i] < \pi[j] \ \wedge \ \sigma[i] > \sigma[j] \ \vee \ \pi[i] > \pi[j] \ \wedge \ \sigma[i] < \sigma[j])\}|$$

The **Kemeny score** is then defined as the sum of Kendall-τ distances between a given permutation and all permutations in a given set. Formally, given any set of permutations $\mathcal{P} \subseteq \mathbb{S}_n$ and a permutation π, we define the Kemeny score S as:

$$S(\pi, \mathcal{P}) = \sum_{\sigma \in \mathcal{P}} D(\pi, \sigma)$$

Finally, the problem of finding an optimal consensus π^* of a set $\mathcal{P} \subseteq \mathbb{S}_n$ is to find π^* such that:

$$\forall \pi \in \mathbb{S} : S(\pi^*, \mathcal{P}) \leq S(\pi, \mathcal{P}).$$

Note that an optimal consensus is not necessarily unique.

Example: Let us consider the rankings r_4 and r_5 in the same dataset as before. Note that both are permutations (with the only difference that letters are used instead of numbers). We can write them the following way: $r4 = [H, A, C, B, G, D, F, E]$, and $r5 = [H, B, A, D, F, C, G, E]$. One optimal consensus ranking of $\mathcal{P}$ is $\pi^* = [H, A, B, D, F, C, G, E]$. The Kemeny score of π^* is made of the pairwise disagreements B-C in r_4 (B is before C in π^* while B is after C in r_4), D-C in r_4, D-G in r_4, F-C in π_1, F-G in r_4, and A-B in r_5 thus $S(\pi^*, \mathcal{P}) = 6$.

Considering the Kendall-τ distance, the rank aggregation problem is known to be NP-hard when the number m of permutations in $\mathcal{P}$ is greater or equal to 4.

2.2 Generalized framework

In real applications, the input rankings may not be permutations but (i) they may have ties, where some elements are at the same position (considered as *ex aequo*) and (ii) they may be incomplete, that is, not all elements are in all rankings. While some works have recently paid attention to the specific case of ranking with ties [6], in the large majority of works incomplete rankings are artificially made complete by using two kinds of techniques: projection (where only common elements to all rankings are considered in the final ranking) and unification as illustrated in the previous section (rankings are completed by adding missing elements at the last positions). Both solutions may lead to unexpected results [6]. More generally, the way of defining a consensus for a set of incomplete rankings may strongly depend on the kind of data and their usage.

c \ r	$x \prec y$	$x \succ y$	$x \equiv y$	$x \urcorner y$	$\urcorner x\ y$	$\urcorner x \urcorner y$
$x \prec y$	0	1	1	0	1	1
$x \equiv y$	1	1	0	1	1	0

Table 1: Generalized Kendall-τ unifying pseudo-distance based costs

c \ r	$x \prec y$	$x \succ y$	$x \equiv y$	$x \urcorner y$	$\urcorner x\ y$	$\urcorner x \urcorner y$
$x \prec y$	0	1	1	0	0	0
$x \equiv y$	1	1	0	0	0	0

Table 2: Generalized induced Kendall-τ measure based costs

Here, we thus introduce a generic framework able to adapt a large range of current consensus ranking algorithms to the general case of possibly incomplete rankings with ties.

We present a score which widely generalizes the Kemeny score defined above, by *(i)* considering incomplete rankings with ties and *(ii)* allowing to tune the parameters in order to adapt to various situations and hypotheses depending on the kind of data and their usage.

This generalized score is precisely defined in [3], we briefly sketch it here. Let $\mathcal{R}$ be a set of (possibly incomplete) rankings with ties on $[n]$. Now suppose that we aim to compute a consensus of the rankings in $\mathcal{R}$. Let c be a ranking with ties containing all elements present in at least one ranking of $\mathcal{R}$. Let x and y be two elements of c. Two situations can occur in c: either x and y are tied (*i.e.*, placed equal), or one is before the other (and in this case we can suppose without loss of generality that x is placed before y). Now let us consider a ranking $r \in \mathcal{R}$. There are six possibilities for x and y in r: (1) x is before y ($x \prec y$); (2) x is after y ($x \succ y$); (3) x and y are tied ($x \equiv y$); (4) x is in r, not y ($x \urcorner y$); (5) y is in r, not x ($\urcorner x\ y$); (6) neither are in r ($\urcorner x \urcorner y$). Accordingly, there are twelve combinations of relative positions of x and y in c and in r. To each of theses combinations we associate a score in $\mathbb{R}$. Finally, the generalized score between c and $\mathcal{R}$ is the sum of all scores between c and the rankings of $\mathcal{R}$, for all pairs of elements x and y. And we define, as usually, an optimal consensus ranking as a ranking $c*$ which minimizes this generalized score.

It is proved in [3] that this new scoring scheme subsumes all scores of the known score-based approaches. Moreover, it also provides the opportunity of designing new measures, finely tuned, to fit a large number of needs in real data. For example, we present the score table of the *Generalized Kendall-τ unifying pseudo-distance* [6] in Table 1 and the score table of the *Generalized induced Kendall-τ measure* [5] in Table 2. The first three columns are identical for the two tables as both measures are equivalent for complete datasets and vary only on incomplete datasets.

Back to the two use cases. In use case 1 (crawlers), the elements which are missing in a given ranking must be considered as less important than the present ones : the following two statements "A is before H" and "A is present and H is missing" must be considered as equivalent. This is why the scores of Table 1 must be used, where the columns $x \prec y$ and $x \urcorner y$ (resp. $x \succ y$ and $\urcorner x\ y$) are the same. This is the Generalized Kendall-τ unifying pseudo-distance.

In use case 2 (rankings of movies), the missing elements have to be interpreted differently: if a movie has not been ranked by a fan, the comparison makes no sense. Here, a missing element must not be considered as less important than a present one. The Generalized induced Kendall-τ measure is the best candidate here: in the cost table 2, the cost is always 0 if x and y are not both present in r.

Thus A is preferred to H in use case 1 while it is the opposite in use case 2, which is the expected behavior since the missing elements have been properly considered.

We have presented two use cases, but users may have specific needs with their datasets. Our new scoring scheme allows to create new scores which will fit better with the specific needs than the current available measures.

3. THE CORANKCO PLATFORM

The CoRankCo platform has been developed in Python 3.5 and uses the Django framework to provide a Web application to users. All datasets, results of runs, and user information are stored in a database. CoRankCo is publicly available at `https://corankco.lri.fr/`[1].

The main interface of CoRankCo is provided in Figure 1 and contains six major panels, allowing users to select algorithms, measures, and datasets to be used for a given run. Note that CoRankCo has been designed to offer a continuum of user profiles from novice (default values provided) to experts (fine tuning of the run). Using the ***Algorithms panel*** users can select which algorithms to run among a set of twelve algorithms whose references can be found in [6]. In CoRankCo, all these algorithms have been reimplemented. Each of these approaches has been generalized, when possible, to incomplete rankings, either by using the generalized framework of Section 2.2 for the score-based approaches (Pick-a-Perm, KwikSort, FaginLarge, FaginSmall, BioConsert, BioCo, CondorcetPartitioning and ExactAlgorithm), or by *ad hoc* adaptations for the positional approaches (BordaCount, CopelandMethod, RepeatChoice and MedRank). In the ***Distance panel***, any *score-based* algorithm can be parameterized with various measures: *Generalized Kendall-τ distance*, *Generalized Kendall-τ distance with unification*, *Induced generalized Kendall-τ distance*, *Generalized Kendall-τ pseudo-distance*. The positional algorithms have been adapted case by case to match with as many measures as possible. If a consensus is requested for an incomplete dataset with a measure and an algorithm such that there is an incompatibility, the user is notified and no result is provided for this request. In the ***Dataset panel***, CoRankCo provides a very large panel of both real and synthetic datasets on which a consensus ranking can be run. The real datasets have been extracted from previous publications (e.g., [6], [10]) and include large datasets about sport competitions (F1, ski), movie preferences, web queries, biology, and Irish elections. In such datasets, the number of rankings and the number of elements to rank may be huge (until several thousands). As for synthetic datasets, CoRankCo provide access to all the datasets used in [6], including datasets with an increasing level of dissimilarity between input rankings. The ***Normalization panel*** allows to choose whether the dataset must be normalized and if so whether it should be unified

[1] Sources are freely available at `https://github.com/bryan-brancotte/rank-aggregation-with-ties`

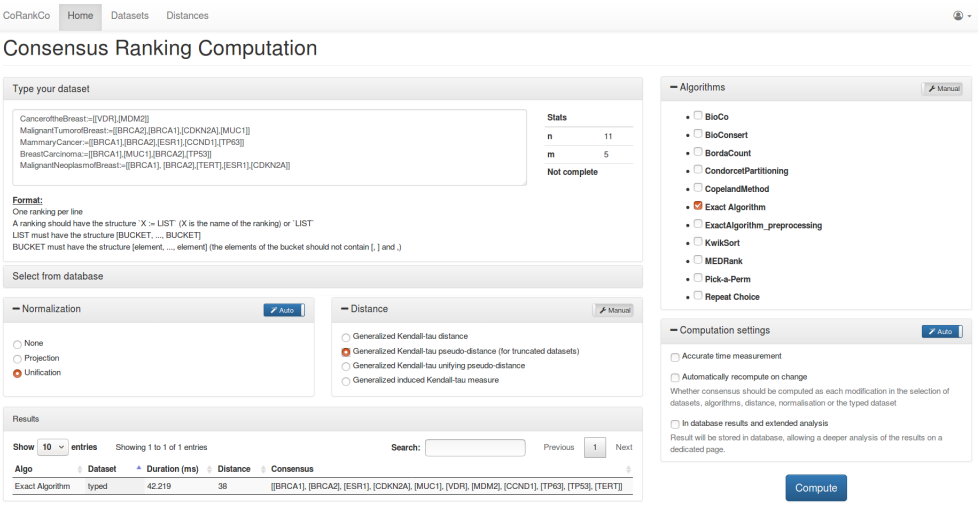

Figure 1: main interface of CoRankCo

(the missing elements in the input rankings are added at the end) or projected (the elements which are not present in each ranking are removed from all the rankings). In the ***Computation settings panel***, several computational settings can be selected. For example, if *Visualization and store results in Database* is set, then the results obtained by several algorithms can be visually compared and stored in the database. The results of a computation are delivered on the ***Results panel***. When several algorithms are run to be compared, visualization of the results obtained (time and distance to the exact algorithms when heuristics are provided) can also be provided.

Last but not least, **users can register to the platform**. In this case they get an account and have several privileges: ability to add their own datasets (making them publicly available or only visible for themselves), create new measures, and save the visualization of the results.

4. DEMONSTRATION

The functionalities of the platform described in the previous section will all be demonstrated. We will make use of three main datasets, namely, on web queries, movies preferences and biological data, to highlight the following keypoints: importance to carefully choose (i) the measure to be used when computing a consensus ranking, (ii) the algorithms or heuristics to use, depending on the data features, notably their size.

Our demonstration will emphasize two points. First, we will consider the biological data of the ConQurBio project [5] and demonstrate the interest of using consensus ranking techniques to help physicians finding relevant results. This part of the demo will focus on ranking genes involved in cancer and cardiac diseases. Second, we will show how CoRankCo allows to reproduce the results obtained by several comparative studies ([6, 2]).

5. ACKNOWLEDGMENTS

This work has been supported in part by CNRS Mastodons.

6. ADDITIONAL AUTHORS

Additional authors: Stéphane Vialette (LIGM, U. Paris Est, Marne la Vallée, France, `vialette@univ-mlv.fr`)

7. REFERENCES

[1] N. Ailon. Aggregation of partial rankings, p-ratings and top-m lists. *Algorithmica*, 57(2):284–300, 2010.

[2] A. Ali and M. Meilă. Experiments with Kemeny ranking: What works when? *Mathematical Social Sciences*, 64(1):28–40, 2012.

[3] P. Andrieu. Ranking aggregation and Kemeny-compatible fonctions. Technical report, 2018.

[4] N. Betzler, R. Bredereck, and R. Niedermeier. Theoretical and empirical evaluation of data reduction for exact Kemeny rank aggregation. *Autonomous Agents and Multi-Agent Systems*, pages 1–28, 2013.

[5] B. Brancotte, B. Rance, A. Denise, and S. Cohen-Boulakia. Conqur-bio: Consensus ranking with query reformulation for biological data. In *Data Integration in the Life Sciences*, volume 8574 of *LNCS*, pages 128–142. 2014.

[6] B. Brancotte, B. Yang, G. Blin, S. Cohen-Boulakia, A. Denise, and S. Hamel. Rank aggregation with ties: Experiments and analysis. *PVLDB*, 8(11):1202–1213, 2015.

[7] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proc. of the 10th World Wide Web conference*, pages 613–622. ACM, 2001.

[8] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing and aggregating rankings with ties. In *Proc. of the 23rd ACM SIGMOD symposium on Principles of database systems*, pages 47–58. ACM, 2004.

[9] M. Jacob, B. Kimelfeld, and J. Stoyanovich. A system for management and analysis of preference data. *Proc. of the VLDB Endowment*, 7(12), 2014.

[10] M. Meila, K. Phadnis, A. Patterson, and J. Bilmes. Consensus ranking under the exponential model. 2007.