Fundamenta Informaticae 158 (2018) 277–295
DOI 10.3233/FI-2018-1649
IOS Press

Collaborative Rankings

Ewa Andrejczuk[*]
Change Management Tool S.L., Barcelona, Spain
Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain
Universitat Autònoma de Barcelona, Bellaterra, Spain
ewa@iiia.csic.es

Juan A. Rodriguez-Aguilar
Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain
jar@iiia.csic.es

Carles Sierra
Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain
sierra@iiia.csic.es

Abstract. In this paper we introduce a new ranking algorithm, called *Collaborative Judgement (CJ)*, that takes into account *peer opinions* of agents and/or humans on objects (e.g. products, exams, papers) as well as *peer judgements* over those opinions. The combination of these two types of information has not been studied in previous work in order to produce object rankings. Here we apply *Collaborative Judgement* to the use case of scientific paper assessment and we validate it over simulated data. The results show that the rankings produced by our algorithm improve current scientific paper ranking practice, which is based on averages of opinions weighted by their reviewers' self-assessments.

Address for correspondence: Campus UAB, 08193 Bellaterra, Spain
[*]Supported by an Industrial PhD scholarship from the Generalitat de Catalunya (DI-060).

1. Introduction

In many areas of our lives we are used to the process of assessing and being assessed. We pass exams at the University, we go through job interviews, we undergo research project reviews, we are evaluated by our employers, etc. Artificial Intelligence research has focused on the assessment process for long and a number of algorithms have been developed to assist in assessing the performance of humans or artificial agents. Indeed, large number of trust and reputation models have been proposed [1, 2, 3, 4, 5].

Surprisingly, to our knowledge, no significant effort has been put in the development of algorithms that use *judgement* information over such assessments. We consider exam marks unjust, interview outcomes biased, and review reports unfair, and we normally comment about these opinions on our performance with friends and relatives. We think that this kind of information is very important as it can be key to build the reputation of assessors. A bad assessor can be detected by the assessing community if they were allowed to simply express their opinions about the bad assessor. Actually, in many social networks this kind of information is collected ("was this recommendation useful to you?"), and presented to users. However, how the sites use this information to rank recommendations is never clearly explained if it is used at all.

Similarly, in the area of multiagent systems, agents' performance is key to build teams and coalitions [6]. Team formation and coalition formation are key for many applications related to multiagent cooperation, e.g. RoboCup rescue team [7, 8], Unmanned Aerial Vehicles (UAVs) operations [9], or team formation in social networks [10] to name just a few. Both team formation and coalition formation focus on assembling the *best* possible group of agents (be it either a team or a coalition) to accomplish some tasks of interest given some limited resources. Hence, it is key for these algorithms to count on an assessment of the *expected capabilities* of the agents to recruit. With this aim, many trust models have been developed in the past to model agent behaviour [11, 12], but judgements have again never been used to our knowledge.

In this paper we present an algorithm, called *Collaborative judgement* (CJ), which wants to go a step further in the use of peer judgements. CJ takes into account judgements on opinions to build reputation values on assessors and then use them to aggregate the opinions of a group of assessors. In current recommender systems the opinions about an object are often aggregated using weights. When no weights are used, the final opinion is usually an average of all the opinions provided (e.g. Amazon or TripAdvisor). When they are used the aggregated opinion is a weighted average using self-assigned weights. This is very common in Conference Management Systems like Confmaster or Easychair. In this paper we will compare CJ with the standard algorithm that weighs opinions with the assessors' self-assessments. We will call this simple algorithm *Self-Assessment Weighted Algorithm* (SAWA).

Here we will particularize the problem of peer judgement to the case of Conference Paper reviewing. The need to improve the way conferences (and to some extent journals) assess papers is key for scientific progress and its pitfalls have been discussed recently, see for instance the NIPS experiment: `http://blog.mrtz.org/2014/12/15/the-nips-experiment.html`. Some researchers have been trying to ameliorate the situation by improving the paper assignment process [13]. However, there is a growing phenomenon in which reviews are not made nor supervised by the expert members of the program committee but by someone to whom the reviewing task is delegated (e.g. a

PhD student). This practice certainly invalidates the potential improvement provided by better assignments. Here we propose to adapt CJ to detect those non-expert reviewers and dismiss their opinions from the final decision on accepting a paper and thus keep the benefits of a better reviewer assignment. Henceforth, the notation we will use will be based on the ontology of a conference: papers, reviewers, marks, etc.

In many settings, including conference paper rankings, the actual numerical value is not the key element but the order between alternatives. Also, this order is not always total as there can be ties between objects (e.g. papers). Therefore, ranking algorithms including CJ have to deal with *partial rankings* between alternatives.

This paper makes the following contributions. First, we define our ranking algorithm based on collective assessments that uses both peer opinions of agents as well as peer judgements over these opinions. We apply it to the case of scientific paper assessment. We compare paper evaluations' accuracy with the currently most used paper evaluation method: the average of opinions weighted by reviewer self-confidence. Finally, we experimentally compare the *partial ranking* among alternatives produced by both methods and the "actual" ranking. The results show that the rankings produced by our algorithm improve those produced with current ranking methods.

The paper is organised as follows. In section 2 we describe the generalisation of the Kendall Tau distance to compare partial rankings. In Section 3 we present the ranking algorithm that we benchmark in Section 6 against SAWA, presented in Section 4. Then, in Sections 7 and 8 we discuss the results and summarise our main achievement and outline our future work.

2. Background: Metrics between partial rankings

In any conference management system, the opinions of reviewers are aggregated to produce a ranking of papers. Notice that this ranking may include ties since several papers may be equally valued. An ordering with ties is also known as a *partial ranking*. Given two different aggregation methods for producing partial rankings, we are interested in comparing them to decide which aggregation method is better. For this purpose, we require metrics to compare partial rankings. The purpose of this section is to introduce such metrics. With this aim we largely rely on the work by Fagin el tal. [14][1] , which provides sound mathematical principles to compare partial rankings. In particular, we will detail one of the four metrics presented in [14], the so-called *Kendall distance with penalty parameter p.* Before that, we require some preliminary definitions.

Definition 2.1. (Bucket order)
A bucket order is, intuitively, a linear order with ties. Formally, given a domain D, a bucket order is a transitive binary relation $\lhd$ for which there are sets $\mathcal{B}_1, \cdots, \mathcal{B}_t$ (the buckets) that form a partition of D such that $x \lhd y$ if and only if there are i, j with $i < j$ such that $x \in \mathcal{B}_i$ and $y \in \mathcal{B}_j$.

A bucket contains objects that are "tied". We say that $\mathcal{B}_i$ is the bucket of x if $x \in \mathcal{B}_i$. We say that bucket $\mathcal{B}_i$ *precedes* bucket $\mathcal{B}_j$ if $i < j$. Thus, $x \lhd y$ if and only if the bucket of x precedes the bucket of y.

[1]We refer the reader to [15] for a more detailed, extended version on the topic.

Notice that a linear order is a bucket order where every bucket is of size 1.

Definition 2.2. (Bucket position)
Let $\mathcal{B}_1, \cdots, \mathcal{B}_t$ be a bucket order over D. The position of a bucket $\mathcal{B}_i$ in the bucket order is defined as $pos(\mathcal{B}_i) = (\sum_{j<i} |\mathcal{B}_j|) + (|\mathcal{B}_i| + 1)/2$.

Intuitively, $pos(\mathcal{B}_i)$ stands for the average location within bucket $\mathcal{B}_i$. Now, we can formally define the notion of partial ranking based on the notion of bucket order.

Definition 2.3. (Partial ranking)
Given a domain D and a bucket order $\mathcal{B}_1, \cdots, \mathcal{B}_t$ over D, the partial ranking σ associated with the bucket order is a function that maps each element in D to the position of its bucket, namely $\sigma(x) = pos(\mathcal{B})$ when $x \in \mathcal{B}$.

Given a partial ranking σ, we say that x *is ahead of* y *in* σ if $\sigma(x) < \sigma(y)$, and that x *and* y *are tied in* σ if $\sigma(x) = \sigma(y)$.

Now, let $\mathcal{P} = \{\{i,j\} | i \neq j \ \ and \ \ i,j \in D\}$ be the set of all the unordered pairs of different elements in D. Given two partial rankings σ_1 and σ_2 with domain D, we will define a penalty measure $\bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2)$ to account for the different ordering of i, j in partial rankings σ_1,σ_2, where p is a fixed parameter such that $0 \leq p \leq 1$. We shall distinguish three cases:

Case 1: i **and** j **are in different buckets in both** σ_1 **and** σ_2**.** (i) If i and j are in the same order in σ_1 and σ_2 (e.g. $\sigma_1(i) > \sigma_1(j)$ and $\sigma_2(i) > \sigma_2(j)$) then $\bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2) = 0$, and thus there is no penalty for $\{i, j\}$. (ii) If i and j are in the opposite order in σ_1 and σ_2 (e.g. $\sigma_1(i) > \sigma_1(j)$ and $\sigma_2(i) < \sigma_2(j)$) then let the penalty $\bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2) = 1$.

Case 2: i **and** j **are in the same bucket in both** σ_1 **and** σ_2**.** Since both partial rankings agree that i and j are tied, there is no penalty and $\bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2) = 0$

Case 3: i **and** j **are in different buckets in only one of the partial rankings.** In this case, the penalty is $\bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2) = p$.

Now we are ready to define the Kendall distance between two partial rankings.

Definition 2.4. (Kendall distance)
Given two partial rankings σ_1 and σ_2 over domain D, we define their $K^{(p)}$, their Kendall distance with parameter p, as follows:

$$K^{(p)}(\sigma_1, \sigma_2) = \sum_{\{i,j\} \in \mathcal{P}} \bar{K}^{(p)}_{i,j}(\sigma_1, \sigma_2).$$

Notice that from the definition above, we can readily define a normalised version of the Kendall distance that we will employ in this paper to compare partial rankings.

Definition 2.5. (Normalised Kendall distance)
Given two partial rankings σ_1 and σ_2 over domain D, their normalised Kendall distance with parameter p is defined as:

$$\tilde{K}^{(p)}(\sigma_1, \sigma_2) = \frac{K^{(p)}(\sigma_1, \sigma_2)}{s}$$

where $s = \frac{|\mathcal{P}|\cdot(|\mathcal{P}|-1)}{2}$ is the number of pairs in $\mathcal{P}$.

Finally, notice that the work in [14] defines three further metrics to compare partial rankings, which also admit efficient computation. However, it does not matter the metric that we choose because the equivalence results in [14] indicate that the four metrics are all within constant multiple of each other.

3. Collaborative judgement

In this section we detail our collaborative judgement algorithm. Before that, we first introduce the notation, which we will use in the rest of the paper.

Definition 3.1. An *Appraisal* is a tuple $\langle P, R, E, o, v\rangle$, where

- $P = \{p_i\}_{i\in\mathcal{P}}$ is a set of objects to be evaluated.
- $R = \{r_j\}_{j\in\mathcal{R}}$ is a set of reviewers.
- $E = \{e_i\}_{i\in\mathcal{E}} \cup \{\bot\}$ is a totally ordered evaluation space, where $e_i \in \mathbb{N}$ and $e_i < e_j$ iff $i < j$ and $\bot$ stands for the absence of evaluation.
- $o : R \times P \to E$ is a function giving the opinions of reviewers on objects.
- $v : R \times R \times P \to E$ is a function giving the judgements of reviewers over opinions on objects.[2] Therefore, a judgement is a reviewer's opinion about another reviewer's opinion.

In general we might have different dimensions of evaluation, that is a number of E spaces over which to express opinions and judgements. For instance, originality, soundness, etc. Nonetheless, here for simplicity reasons we will assume that the evaluation of an object is made over a single dimension. Actually, the 'overall' opinion is what is aggregated in real systems.

The steps of the CJ algorithm applied over an appraisal $\langle P, R, E, o, v\rangle$ are as follows:

Step 1. Compute the *agreement level* between each pair of reviewers r_i and r_j as a function $a : R\times R \to [0,1]\cup\{\bot\}$. This computation involves the set of objects jointly reviewed by reviewers r_i and r_j, which we will formally define as $P_{ij} = \{p_k \in P | o(r_i, p_k) \neq \bot, o(r_j, p_k) \neq \bot\}$. If two reviewers jointly reviewed objects, then their agreement level is based on the similarities of

[2]In tools used to evaluate papers like ConfMaster (`www.confmaster.net`), this information could be gathered by simply adding a private question to each paper review, answered with elements in E, one value in E for the judgement on each fellow reviewer's review.

their opinions on common objects as well as on their judgements. Formally, we compute the agreement level as:

$$a(r_i, r_j) = \begin{cases} \frac{\sum_{p_k \in P_{ij}} s(r_i, r_j, p_k)}{|P_{ij}| \cdot d} & \text{if } P_{ij} \neq \emptyset \\ \perp & \text{otherwise} \end{cases} \quad (1)$$

where d is the maximum distance in the evaluation space and:

$$s(r_i, r_j, p_k) = \begin{cases} v(r_i, r_j, p_k) & \text{if } P_{ij} \neq \emptyset \text{ and } v(r_i, r_j, p_k) \neq \perp \\ Sim(o(r_i, p_k), o(r_j, p_k)) & \text{if } P_{ij} \neq \emptyset \text{ and } v(r_i, r_j, p_k) = \perp \\ \perp & \text{otherwise} \end{cases} \quad (2)$$

Sim stands for an appropriate similarity measure. When no explicit judgements are given, we use the similarity between opinions as a heuristic of their values. This is based on the following assumption: the more similar a review is to my opinion, the better I am bound to judge that opinion.

Step 2. Compute a complete *Trust Graph* as an adjacency function matrix $C = \{c_{ij}\}_{i,j \in R}$ such that:

$$c(r_i, r_j) = \begin{cases} a(r_i, r_j) & \text{if } a(r_i, r_j) \neq \perp \\ \max\limits_{h \in chains(r_i, r_j)} \prod\limits_{(k,k') \in h} a(r_k, r_{k'}) & \text{otherwise} \end{cases} \quad (3)$$

where $chains(r_i, r_j)$ is the set of sequences of reviewer indexes connecting i and j. Formally, a chain h between reviewers i and j is a sequence $\langle l_1, \ldots, l_{n_h} \rangle$ such that $l_1 = i$, $l_{n_h} = j$, and $a(r_k, r_{k+1}) \neq \perp$ for each pair $(k, k+1)$ of consecutive values in the sequence. To compute this step we use a version of Dijkstra's algorithm that instead of looking for the shortest path (using $+$ and $\min$ as mathematical operations), it looks for the path with the largest edge product (using $\cdot$ and $\max$ as mathematical operators). The running time of the Dijkstra algorithm can take $O(n \log n)$, where $n = |R|$, if using priority queues [16].

Step 3. Compute a *reputation* for each reviewer in R, $\{t_i\}_{i \in R}$. With this aim, we follow the notion of transitive trust: If a peer i trusts any peer j, it would also trust the peers trusted by j. Since this principle is employed by the Eigentrust algorithm [17], we use it to compute reviewer reputations. The use of Eigentrust allows us to obtain a global trust value for each reviewer by the repeated and iterative multiplication and aggregation of reputation values until the trust grades for all agents converge to stable values. Note that the trust graph generated in step 2 is aperiodic and strongly connected as required by the Eigentrust algorithm. Furthermore, we normalise the powers of the matrix C at each step to ensure its convergence. In vectorial notation, the trust vector is assessed as $\bar{t} = \lim_{k \to \infty} \bar{t}^{k+1}$ with $\bar{t}^{k+1} = C^T \bar{t}^k$ and $\bar{t}^0 = \bar{e}$ being $\bar{e}_i = 1/|\bar{e}|$. The complexity of the Eigentrust algorithm used in this step is $O(|R|^2)$.

Step 4. Compute the *collective opinion* on each object as a weighted average of the opinions of those that expressed an opinion on the object. In other words, given an object p_j, we only consider the opinions of those reviewers that reviewed p_j, which we formally define as $R_j \subseteq R, R_j = \{r \in R | o(r, p_j) \neq \perp\}$. We can then compute the collective opinion on an object p_j as a weighted average of the opinions of the reviewers in R_j using as weights the reviewers' reputations. Finally, the collective opinion computed by our collaborative judgement algorithm for an object p_j, noted as $o_{CJ}(p_j)$, is:

$$o_{CJ}(p_j) = \frac{\sum_{r \in R_j} \bar{t}_r \cdot o(r, p_j)}{\sum_{r \in R_j} \bar{t}_r} \tag{4}$$

where $\bar{t}_r$ stands for the reputation value of reviewer r.

Step 5. Generate a partial ranking based on the set of collective opinions $O_{CJ}(P)$. CJ sorts objects in descending order by the collective opinion values. Thus, the object with the highest value of collective opinion gets the first ranking position. Objects with equal collective opinion receive the same ranking number, and the object(s) on the next position receive the immediately following ranking number (i.e. bucket index). The procedure continues until CJ assigns bucket indexes to all objects.

4. The self-assessment weighted algorithm

A conference management system is a web-based application that supports, *inter alia*, the evaluation and selection of articles for scientific purposes (mainly conferences and to some degree journals). The most common approach to paper evaluation used in systems such as Confmaster or Easychair is as follows:

1. Assign every article to (normaly) three reviewers based on either keywords distinguished by using word frequency analysis, and eventually their preferences expressed as bids.
2. Ask each reviewer to assess (give an opinion on) each of their assigned papers and also assess their own confidence on each evaluation.
3. Determine the overall opinion on each paper as a weighted average of the opinions of the reviewers of the paper considering their *self-assessed confidences* as weights.
4. Build a (partial) ranking of articles based on the overall opinions.

We will refer to the algorithm above as the *self-Assessment Weighted Algorithm* (SAWA). Next, we formalise how step 3 in SAWA computes the overall opinion on each paper. We assume that a function $\kappa : R \times P \mapsto [0, 1]$ keeps how confident each reviewer feels about their opinion on a paper. Then, given a paper p_j evaluated by a set of reviewers R_j, SAWA computes the aggregated opinion on the paper as:

$$o_{SAWA}(p_j) = \frac{\sum_{r \in R_j} \kappa(r, p_j) \cdot o(r, p_j)}{\sum_{r \in R_j} \kappa(r, p_j)} \tag{5}$$

We rank articles in descending order according to values $O_{SAWA}(P)$ in the same way as in CJ.

5. Motivating example

The purpose of this section is to illustrate how the CJ and SAWA algorithms described in sections 3 and 4 work to produce paper rankings. Before that, we introduce some matrix notation that will help us describe CJ's operation in a concise manner. Thus, let $O : |P| \times |R|$ be the opinion matrix; $A : |R| \times |R|$ be the agreement level matrix; $V_k : |R| \times |R|$ the individual judgement matrix for paper p_k containing only direct judgements of reviewers; $S_k : |R| \times |R|$ the judgement matrix for paper p_k; $C : |R| \times |R|$ the trust matrix; and $\bar{t}$ the reputation vector.

Now, say that there are only four papers to be reviewed $P = \{p_0, p_1, p_2, p_3\}$ and four reviewers $R = \{r_0, r_1, r_2, r_3\}$ available to give their opinions on papers. Our objective is to choose two top-rated articles out of P and compare CJ and SAWA rankings of the papers in P. We consider that reviewers r_0 and r_1 are qualified, which means that they can recognize the value of a paper and rate it adequately. Reviewers r_2 and r_3 provide unfair opinions as they are incompetent, but they can distinguish between a good and a bad review, namely they can judge correctly. Every article is assigned to three reviewers as follows:

- p_0 is assigned to $\{r_0, r_1, r_2\}$,
- p_1 is assigned to $\{r_1, r_2, r_3\}$,
- p_2 is assigned to $\{r_1, r_2, r_3\}$,
- p_3 is assigned to $\{r_0, r_1, r_3\}$

We assume that all reviewers but r_3 complete their reviews. Reviewer r_3 did not evaluate article p_3. Based on the collected reviews, the opinion matrix O looks as follows:

$$O = \begin{bmatrix} 0.1 & 0.1 & 0.7 & \perp \\ \perp & 0.5 & 0.5 & 0.9 \\ \perp & 0.6 & 0.7 & 0.4 \\ 0.9 & 0.9 & \perp & \perp \end{bmatrix}$$

For instance, the opinion of reviewer r_3 on paper p_2 is 0.4, namely the value of $O[2, 3]$.

Besides reviews, each reviewer evaluates their own confidence on each of their reviews and judges the reviews of other reviewers whenever they have papers in common. In other words, given a paper p_k, the reviewers in R_k judge one another. Thus, each individual judgement matrix V_k will contain each reviewer self-assessment together with the reviewers' judgements on other reviews of p_k. Say that the individual judgement matrices in our example are defined as follows:

$$V_0 = \begin{bmatrix} 0.9 & 1.0 & 0.8 & \perp \\ 1.0 & 1.0 & 0.8 & \perp \\ 0.2 & 0.2 & 0.7 & \perp \\ \perp & \perp & \perp & \perp \end{bmatrix} \quad V_1 = \begin{bmatrix} \perp & \perp & \perp & \perp \\ \perp & 0.9 & 1.0 & 0.6 \\ \perp & 0.3 & 1.0 & 0.7 \\ \perp & 0.2 & 0.7 & 0.8 \end{bmatrix}$$

$$V_2 = \begin{bmatrix} \perp & \perp & \perp & \perp \\ \perp & 0.9 & 1.0 & 0.7 \\ \perp & 0.5 & 1.0 & 0.8 \\ \perp & 0.3 & 0.1 & 0.6 \end{bmatrix} \quad V_3 = \begin{bmatrix} 1.0 & 0.9 & \perp & \perp \\ \perp & 1.0 & \perp & \perp \\ \perp & \perp & \perp & \perp \\ \perp & \perp & \perp & \perp \end{bmatrix}$$

For instance, consider the individual judgement matrix V_0. Reviewer r_2 indicates that their self-assessed confidence on their review of paper p_0 is 0.7, namely the value of $V_0[2,2]$. Furthermore, reviewer r_2 judges the review made by r_1 with a 0.2 value, namely the value of $V_0[2,1]$.

At this point, we count on all the input information required by CJ and SAWA to perform paper assessment and produce paper rankings. Next, in sections 5.1 and 5.2 we illustrate CJ's and SAWA's operations respectively, while in section 5.3 we compare the rankings produced by both algorithms.

5.1. The collaborative judgement algorithm at work

We follow the steps for the CJ algorithm described in section 3:

Step 1. Compute the *agreement level* between reviewers. This requires that we compute first the judgement matrices S_0, S_1, S_2, and S_3 for the papers in P using equation 2. CJ sets $S_0 = V_0$, $S_1 = V_1$, and $S_2 = V_2$. As to V_3, it finds that there is a missing judgement of r_0 about r_1 when it comes to opinion about paper p_3. Then, it calculates this missing judgement by considering the difference in opinions between r_0 and r_1 on paper p_3 using the following similarity measure:

$$Sim(o(r_i, p_k), o(r_j, p_k)) = 1 - |o(r_i, p_k) - o(r_j, p_k)|$$

Hence, the final matrix of judgements for paper p_3 looks as follows:

$$S_3 = \begin{bmatrix} 1.0 & 0.9 & \perp & \perp \\ 1.0 & 1.0 & \perp & \perp \\ \perp & \perp & \perp & \perp \\ \perp & \perp & \perp & \perp \end{bmatrix}$$

Now CJ can employ equation 1 to calculate the agreement level matrix:

$$A = \begin{bmatrix} 0.95 & 0.95 & 0.80 & \perp \\ 1.00 & 0.95 & 0.93 & 0.65 \\ 0.20 & 0.33 & 0.90 & 0.75 \\ \perp & 0.25 & 0.40 & 0.70 \end{bmatrix}$$

Step 2. Compute a complete *trust graph*. If we look at equation 3, we observe that we can readily obtain most trust values from the agreement matrix A. In fact, we only miss the trust values between r_0 and r_3 (notice that $a[0,3] = \perp$ and $a[3,0] = \perp$ in the agreement matrix A). Therefore, CJ only has to compute $c(r_0, r_3)$ and $c(r_3, r_0)$. Recall from section 3 that the missing trust value for a pair or reviewers r_i and r_j is computed by finding the chain (path) of reviewers connecting r_i and r_j with maximum trust product. Figure 1 shows a graph-based representation of the agreement level matrix A, nodes stand for reviewers and a directed edge from r_i to r_j is labeled with the agreement level between r_i on r_j, namely $a(r_i, r_j)$.

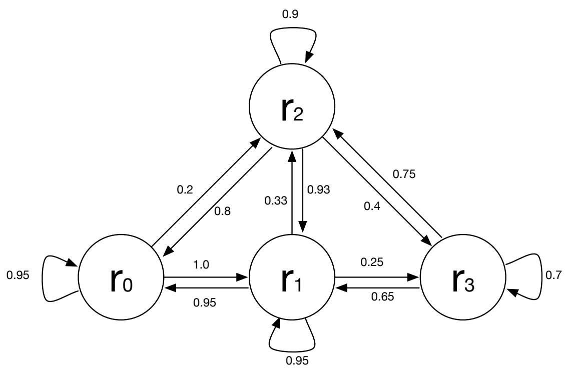


Figure 1: The graph representing the agreement level between reviewers.

Our algorithm finds that the missing agreement levels between r_3 and r_0 are:

- $c(r_0, r_3) = 1.0 \cdot 0.25 = 0.25$ because the chain with maximum trust product is $\langle r_0, r_1, r_3 \rangle$
- $c(r_3, r_0) = 0.75 \cdot 0.93 \cdot 0.95 = 0.663$ because the chain with maximum trust product is $\langle r_3, r_2, r_1, r_0 \rangle$.

By putting together the values of the agreement level matrix and the missing agreement levels $c(r_0, r_3)$ and $c(r_3, r_0)$, we finally obtain the trust matrix C:

$$C = \begin{bmatrix} 0.95 & 0.95 & 0.80 & 0.66 \\ 1.00 & 0.95 & 0.93 & 0.65 \\ 0.20 & 0.33 & 0.90 & 0.75 \\ 0.25 & 0.25 & 0.40 & 0.70 \end{bmatrix}$$

Step 3. Compute a *reputation* value for each reviewer in R by using Eigentrust. Finally, the algorithm computes the reputation values of each agent by applying Eigentrust with the C matrix obtained at step 2 as an input.[3] Eigentrust converges to the following reputation vector:

$$\bar{t} = \begin{bmatrix} 0.344 & 0.358 & 0.169 & 0.129 \end{bmatrix}^T$$

[3]The matrix gets transposed to be used by the Eigentrust algorithm, therefore each reviewer reputation is represented by one column

Each row in $\bar{t}$ represents a reputation value for each one of the reviewers in R. We observe that the reputations of r_2 and r_3 are 0.169 and 0.129 respectively. Therefore, CJ found that these two reviewers are not competent.

Step 4. Compute the *collective opinion* on objects as a weighted average of the opinions of those that expressed an opinion. Having assessed the agents' reputation, CJ can calculate the collective opinion for each paper using equation 4. The resulting opinions for each paper are shown in figure 2.

Step 5. Generate a partial ranking based on the set of collective opinions $O_{CJ}(P)$. CJ generates a paper ranking that comes from ordering papers according to the opinion values in descending order, as shown in figure 3.

5.2. SAWA at work

SAWA computes the opinion on each paper by combining the values in the opinion matrix O with the self-assessed confidence of each reviewer in the individual judgement matrices V_0, V_1, V_2, and V_3, namely with the value in the diagonals of these matrices. For instance, let us calculate the opinion on article p_2. This requires the opinions of reviewers r_1, r_2, and r_3 on the article ($O[2,1] = 0.6$, $O[2,2] = 0.7$, and $O[2,3] = 0.4$). It also requires the confidence values of those reviewers in their reviews, which are contained in matrix V_2: $V_2[1,1] = 0.9$, $V_2[2,2] = 1.0$, and $V_2[3,3] = 0.6$ are the confidence values of r_1, r_2, and r_3 respectively on their own reviews on p_2. Now, using equation 5, SAWA assesses the opinion on p_2 as a weighted average of opinions using confidence values as follows:

$$o_{SAWA}(p_2) = \frac{0.6 \cdot 0.9 + 0.7 \cdot 1.0 + 0.4 \cdot 0.6}{0.9 + 1.0 + 0.6} = 0.592$$

The opinions for the rest of articles are shown in Figure 2 below.

5.3. Comparing rankings

Next we compare the paper rankings produced by CJ and SAWA with the ranking resulting from an "oracle" that knows the true quality of the papers. Figure 3 shows the produced rankings based on the opinions in Figure 2.

	p0	p1	p2	p3
CJ	0.217	0.579	0.586	0.900
SAWA	0.262	0.619	0.592	0.900
True Quality	0.100	0.500	0.600	0.900

Figure 2: Opinions obtained by CJ and SAWA per paper together with their true quality values.

	Ranking
CJ	{p3}, {p2}, {p1}, {p0}
SAWA	{p3}, {p1}, {p2}, {p0}
True Ranking	{p3}, {p2}, {p1},{p0}

Figure 3: Ranking produced by CJ and SAWA along with the ranking resulting from the papers' true qualities.

We observe that the ranking produced by CJ is the same as the oracle's, while SAWA yields a different ranking. This is because CJ exploited judgement information to find out that reviewers r_2 and r_3 are bad reviewers (their reputation values are the lowest ones in $\bar{t}$). This reduced the significance of their opinions when evaluating article p_1, and also increased the importance of the opinion of reviewer r_1, who is a good reviewer. As a result, the opinion on p_2 is larger than the opinion on p_1. Contrarily, SAWA valued article p_1 better than p_2. This is because reviewer r_3, a bad reviewer, evaluated better p_1 than p_2 and reported a high self-assessed confidence value. As a result, p_1's opinion outperformed p_2's despite p_2 true quality is higher.

Our example tells us that a more informed algorithm (by adding judgements of opinions) helped us discriminate good assessments from bad assessments. By all means this is just a toy example intended to illustrate our algorithm. In what follows we perform a more substantial evaluation.

Before that, notice that our example only considered full rankings instead of partial rankings (rankings with ties) to ease comprehension.

6. Experimental evaluation

The purpose of this section is to evaluate the CJ algorithm via simulation. With this aim, we benchmark CJ and SAWA against an "oracle" that knows the true quality of papers. Our analysis will measure:

- the accuracy of the opinions and rankings produced by CJ and SAWA;
- the robustness of CJ against bad reviewers; and
- the sensitivity of global trust to bad reviewers.

Our study will confirm that CJ is the algorithm of choice to compute rankings on objects taking into account peer opinions. Next, in section 6.1 we formulate the hypothesis that our experiments pursue to validate. Section 6.2 describes our experimental settings and section 6.3 dissects the results of the three experiments providing support to our hypotheses.

6.1. Hypotheses

In order to demonstrate that CJ is the algorithm of choice to compute rankings taking into account peer opinions, the experiments that follow focus on validating the next hypotheses:

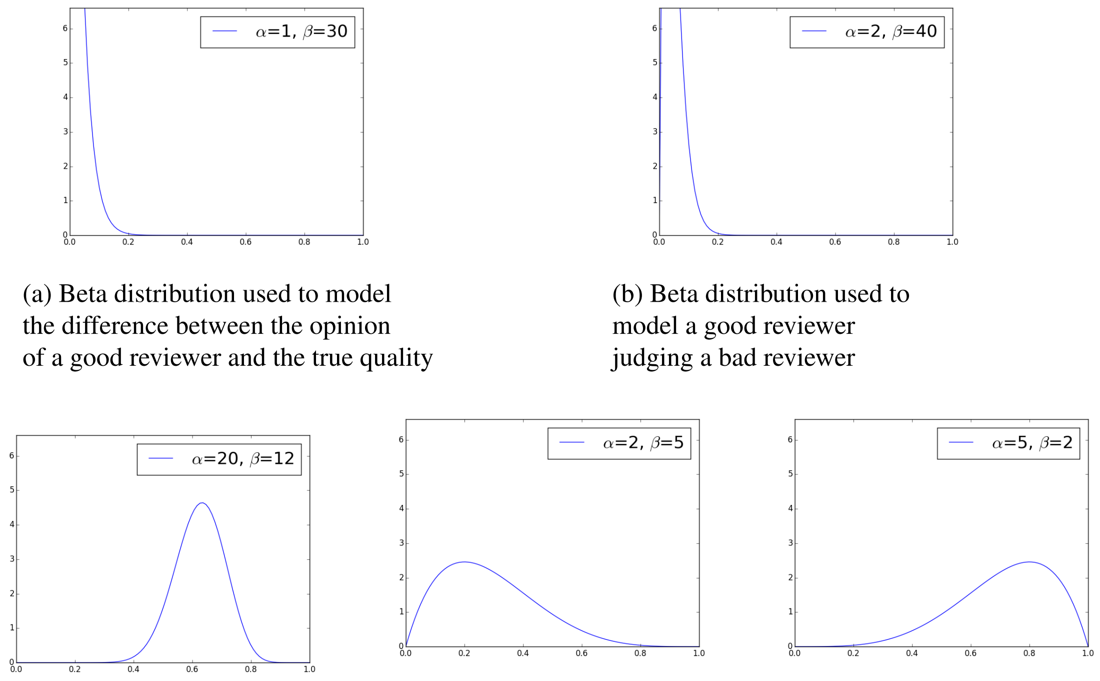
H1 **CJ evaluations get closer than SAWA's to the true quality of a paper as the number of good reviewers increase.**[4]

H2 **The rankings produced by CJ get closer to the true ranking than SAWA's as the number of good reviewers increase.**

H3 **Ceteris paribus, the better the reviewers, the better the accuracy (in terms of opinions and rankings) of CJ with respect to SAWA.**

6.2. Experimental settings

We assume a set $P = \{p_1, \ldots, p_n\}$ of papers and a function for their true quality in a range $[0, 1]$,[5] $q : P \to [0, 1]$. We use the following evaluation space $E = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$, which is rather common in the context of paper reviewing.



(a) Beta distribution used to model the difference between the opinion of a good reviewer and the true quality

(b) Beta distribution used to model a good reviewer judging a bad reviewer

(c) Beta distribution used to model a bad reviewer opinion

(d) Beta distribution used to model a bad reviewer judging a good reviewer

(e) Beta distribution used to model a bad reviewer judging a bad reviewer

Figure 4: Beta Distribution for different configurations of α and β parameters

[4]See next subsection for our representation of a good reviewer.

[5]Assessing the true quality of an object may be difficult and it is certainly a domain dependent issue.

We use beta distributions to model reviewers' opinions and judgements as it is an appropriate distribution to simulate a behaviour that is subject to random variation and is limited on both extremes, i.e. represents processes with natural lower and upper boundaries [18]. Depending on the α and β parameters, the shape of the beta distribution changes substantially (see figure 4 below with different configurations of both variables).

We model two types of reviewers: good and bad, with the following behaviour:

- *Good reviewer.* She provides fair opinions and fair judgements. Her opinion on any paper p_k is always close to its true quality $q(p_k)$. We assume that the absolute value of the difference between the opinion of a reviewer and the true quality of a paper (as a percent) follows a beta distribution, $Beta(\alpha, \beta)$, very positively skewed, for instance with $\alpha = 1$ and $\beta = 30$. For each paper p_k reviewed by a good reviewer, we sample the reviewer's associated beta distribution for a percentage difference, apply it to the paper quality $q(p_k)$ (up or down randomly) and round the result to fit an element in E. Her judgements on someone's opinion are close to 0 if that opinion is far from the true quality of the paper, and close to 1 otherwise. We implement this as the following function:

  $$v(r_i, r_j, p_k) = 1 - |o(r_j, p_k) - q(p_k)|$$

  and self-judgements from $Beta(5, 2)$, slightly negatively skewed.

  We assume that when a good reviewer judges a bad reviewer she samples a value in E from a beta distribution rather positively skewed: $Beta(2, 40)$. The intuition is that good reviewers poorly mark bad reviews.

- *Bad reviewer.* She provides unfair opinions, because she is incompetent, but provides reasonable judgements as she can interpret the opinions of others as being informative or not. Thus, we sample opinions from $Beta(20, 12)$ —rather central with a slight negative skew, judgements for good reviews and self-judgements from $Beta(5, 2)$ as for good reviewers —negatively skewed, and judgements on bad reviews from $Beta(2, 5)$ —slightly positively skewed. The overall idea is that bad reviewers stay mostly in the central area of the evaluation space.

We use $Sim(x, y) = (|E| - 1 - |\tau(x) - \tau(y)|)/(|E| - 1)$ as a simple linear similarity function where τ is a function that gives the position of an element in the ordered set E.

6.3. Results

In this section we present our experimental results using the settings described above.

6.3.1. Analysing the accuracy of opinions

Here we consider the accuracy of a collective opinion on a paper as the difference between that opinion and the true quality of the paper. Then we compare the accuracy of the opinions computed by CJ and SAWA as the percentage of good reviewers increases. We compute the accuracy of both CJ and

SAWA as the mean absolute error of their opinions with respect to the true qualities using the following expressions:

$$MAE_{CJ} = \frac{\sum_{p \in P} |o_{CJ}(p) - q(p)|}{|P|} \qquad MAE_{SAWA} = \frac{\sum_{p \in P} |o_{SAWA}(p) - q(p)|}{|P|}$$

where q is a function that yields the true quality of each paper. Figure 5 plots the percentage error reduction of CJ with respect to SAWA (computed as $(1 - \frac{MAE_{CJ}}{MAE_{SAWA}}) \cdot 100$) by aggregating the values obtained from 30 runs of each algorithm (each run samples all the distributions and thus generates different collective assessments). Note that CJ outperforms SAWA, as it is much more resilient to bad reviewers. As a matter of fact, as opposed to SAWA that treats all reviewers equally, CJ is designed to detect bad reviewers and diminish the importance of their opinions by the usage of the reputation measure. We observe that CJ's gains become larger than 20% and statistically significant for percentages of good reviewers between 20% and 80%. Therefore, these results support hypotheses H1.

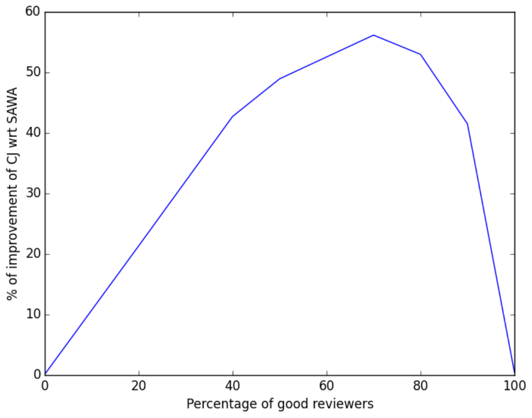


Figure 5: Accuracy of opinions: percentage of error improvement of CJ over SAWA.

6.3.2. Analysing the accuracy of rankings

Now we compare the accuracy of the rankings produced by CJ and SAWA with respect to the ranking resulting from the true quality of papers. In order to compare two partial rankings we employ the normalised Kendall distance in definition 2.5 with penalty factor $p = 0.5$. We employed the partial rankings resulting from 30 runs of CJ and SAWA. We note by $\sigma_1^{CJ}, \dots, \sigma_{30}^{CJ}$ the partial rankings produced by CJ by $\sigma_1^{SAWA}, \dots, \sigma_{30}^{SAWA}$ the partial rankings produced by SAWA, and by σ^q the true ranking. Then, for each partial ranking computed by CJ and SAWA, we compute its normalised Kendall distance with respect to the true ranking. On the one hand, we assess the average Kendall distance of the rankings produced by CJ as $K_{CJ} = \frac{\sum_{i=1}^{30} \tilde{K}^{(0.5)}(\sigma_i^{CJ}, \sigma^q)}{30}$. On the other hand, we assess the average Kendall distance of the rankings produced by SAWA as $K_{SAWA} = \frac{\sum_{i=1}^{30} \tilde{K}^{(0.5)}(\sigma_i^{SAWA}, \sigma^q)}{30}$.

Figure 6 (left) plots the average Kendall distance of the rankings produced by CJ with respect to the true ranking, namely K_{CJ}, as the number of good reviewers increases. We observe that the distance between CJ rankings and the true ranking quickly decreases as the number of good reviewers increases. Notice that beyond 50% of good reviewers the distance drops below 0.1. That means that CJ can produce rather accurate rankings despite the presence of a large ratio of bad reviewers.

Figure 6 (right) shows the accuracy gain of CJ with respect to SAWA. We calculate such accuracy gain as $\frac{K_{SAWA}-K_{CJ}}{K_{SAWA}} \cdot 100$. We observe that the accuracy gain yield by CJ as the number of good reviewers grows, going beyond a 40% gain with 80% good reviewers. Similarly to experiment 6.3.1, the graph clearly shows that CJ performs significantly better even when the number of bad reviewers is high. We see that CJ has been able to discriminate poor assessments, while SAWA treats all reviews equally. We observe also that CJ benefits larger from good reviewers than SAWA.

The results depicted in Figure 6 support hypothesis H2.

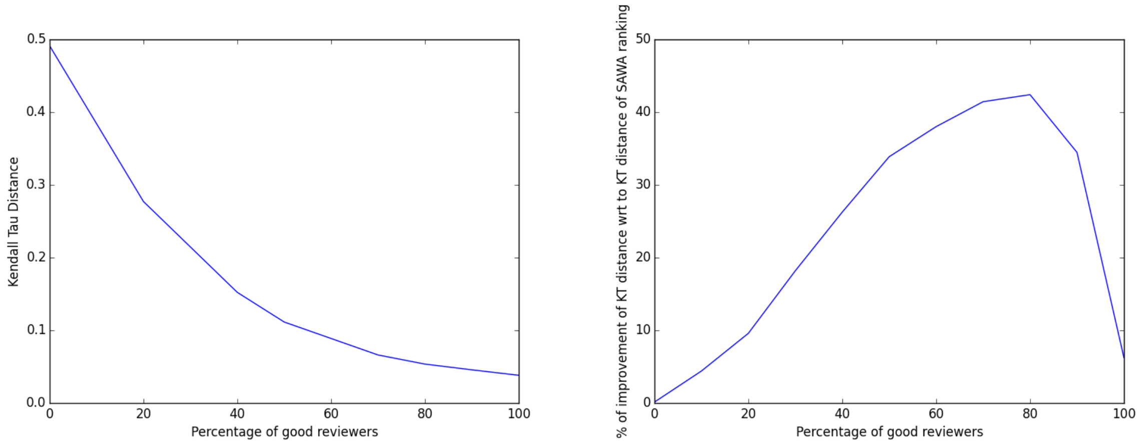


Figure 6: (Left) Normalised Kendall Ranking distance calculated for CJ ranking and true ranking of the papers. (Right) Percentage of error decrease measured as a Kendall distance between rankings produced by CJ and SAWA and true ranking of papers for increasing percentages of good reviewers.

6.3.3. Analysing the robustness against bad reviewers

As mentioned before, we model good reviewers' opinions with a $Beta(\alpha, \beta)$ very positively skewed from which we sample the difference between the reviewer's opinion and the true quality. With $\alpha = 1$ and $\beta > 30$ the expert is frequently telling the true quality in her opinions (specially because we discretise the sampled values into our evaluation space —i.e. almost all the distribution mass is rounded to a distance of 0 with respect to the true quality). In figure 7 we plot the improvement of CJ with respect to SAWA for $\alpha = 1$ and increasing values of β (better reviewer behaviour). We observe that the algorithm outperforms SAWA by 10% when reviewer is frequently mistaken ($\beta = 5$). This shows that even when good reviewers give frequently inaccurate opinions, CJ is still able to capture them and increases the importance of their assessments. The improvement asymptotically grows to 51% with increasing quality of the reviewer behaviour. These results support Hypothesis H3.

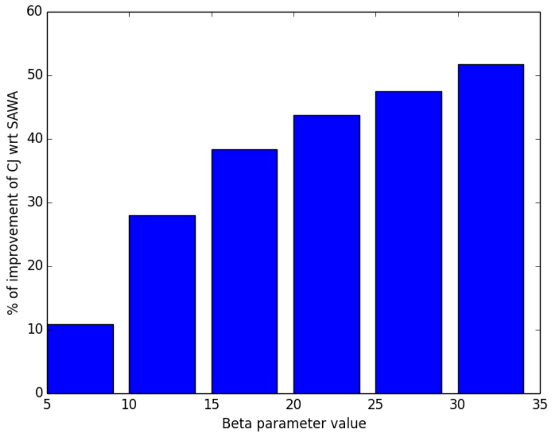


Figure 7: Improvement of CJ over SAWA as the reviewers' quality increases (with fixed $\alpha = 1$ and increasing β values). This plot is for a population with 50% good reviewers and 50% bad reviewers.

7. Discussion

One issue worth discussing is the feasibility of getting real data to model $q(\cdot)$. We mentioned before that this is obviously a domain dependent issue and that it can be difficult to obtain. In the case of paper review, what is the true quality of a paper? It seems impossible to answer this question. We could get data on impact of papers and assume that impact relates to quality. This can be done for the papers that were accepted and published, but not for those that were rejected. Therefore, the validation of the algorithm results will necessarily be partial. This will always be controversial as the use of any quality metric would always be debatable. It is in this context that our algorithm contributes since the key assumption of our algorithm is: *when there is no clear-cut method to determine the quality of an object, then the true quality can be determined by the social acceptance of the opinions expressed by experts*. Hence, the use of the *best* experts' ranking can be understood as the ranking of the socially most *reputed* experts. Precisely what CJ aims at modelling.

In terms of scalability, the current version of CJ uses Dijstra's algorithm and matrix operations that scale up reasonably well (quadratically), but there are improvements that can be done by distributing the computation as in some versions of Eigentrust.

Another issue worth mentioning is that reviewer quality depends on the particular subarea of a conference. In general, our opinions are more or less fair depending on our true competences. Thus, CJ should consider this dimension as many existing trust models do [19, 6]. The inclusion of a semantic dimension on trust and reputation requires defining an ontology of the domain and semantic distances between the elements in the vocabulary. This represents no technical problem and will basically increase the complexity of the computation proportionally to the granularity of the vocabulary.

Malicious agents can collude to artificially over rate their works. Eigentrust has extensions that are robust against this collusion and can be used as an improvement of CJ [17].

8. Conclusions and further work

In this paper we introduced CJ. It is a new ranking algorithm that takes into account *peer opinions* of agents and/or humans as well as *peer judgements* over those opinions. We applied CJ to the use case of scientific paper assessment and we validated it over simulated data. The results show that the rankings produced by this new algorithm (under (reasonable) assumptions on reviewer behaviour) improve current scientific paper ranking practice. The use of this algorithm in the context of agent team formation is key as it will provide a sound method to assess the *capabilities* of agents by observing peer opinions and judgements made by agents and humans.

Part of the future work is centred on evaluating CJ over real data. We are planning to get data from a commercial bank about the skills of team members. That is, employees work in teams to solve tasks that require specific skills. Team members record opinions on their team-mates' skills and judgements on the opinions after anonymisation. We are also discussing the extension of functionalities of a major conference management system so that we can get data on judgements in conferences in the near future.

At simulation level we want to further explore the sensitiveness of the results for varying parameter settings, including the impact of different similarity functions. Finally, the modelling of *malicious* reviewers (those who know the quality of a paper and deliberately lie about it) will be considered. We expect that our method might help in detecting those reviewers.

Finally, this algorithm is an important milestone on our path to develop methods to build agent and human teams to solve complex tasks that balance capabilities and mutual relationships.

Acknowledgments: This work is supported by the CollectiveMind project (Spanish Ministry of Economy and Competitiveness, grant number TEC2013-49430-EXP), the COR (TIN2012-38876-C02-01), Collectiveware projects (TIN2015-66863-C2-1-R) and MILESS (Ministerio de Economa y Competitividad) TIN2013-45039-P Financed by FEDER.

References

[1] Piech C, Huang J, Chen Z, Do C, Ng A, Koller D. Tuned Models of Peer Assessment in MOOCs. *Proc. of the 6th International Conference on Educational Data Mining (EDM 2013)*, 2013. arXiv:1307.2579 [cs.LG].

[2] de Alfaro L, Shavlovsky M. Crowdgrader: Crowdsourcing the evaluation of homework assignments. *Tech. Report 1308.5273, arXiv.org*, 2013. doi:10.1145/2538862.2538900.

[3] Walsh T. The PeerRank Method for Peer Assessment. In: Schaub T, Friedrich G, O'Sullivan B (eds.), ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014), volume 263 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. 2014, pp. 909–914. ISBN 978-1-61499-418-3.

[4] Wu J, Chiclana F, Herrera-Viedma E. Trust based consensus model for social network in an incomplete linguistic information context. *Applied Soft Computing*, 2015;35:827–839. `https://doi.org/10.1016/j.asoc.2015.02.023`.

[5] Zhang J, Ghorbani AA, Cohen R. A familiarity-based trust model for effective selection of sellers in multiagent e-commerce systems. *Int. J. Inf. Sec.*, 2007;6(5):333–344. doi:10.1007/s10207-007-0025-y.

[6] Osman N, Sierra C, McNeill F, Pane J, Debenham JK. Trust and matching algorithms for selecting suitable agents. *ACM TIST*, 2013;5(1):16. doi:10.1145/2542182.2542198.

[7] Ramchurn SD, Farinelli A, Macarthur KS, Jennings NR. Decentralized Coordination in RoboCup Rescue. *Comput. J.*, 2010;53(9):1447–1461. doi:10.1093/comjnl/bxq022.

[8] Nair R, Tambe M, Marsella S. Team Formation for Reformation in Multiagent Domains Like RoboCupRescue. In: Kaminka G, Lima P, Rojas R (eds.), RoboCup 2002: Robot Soccer World Cup VI, volume 2752 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 2003 pp. 150–161. ISBN: 978-3-540-40666-2.

[9] Haque M, Egerstedt M, Rahmani A. Multilevel Coalition Formation Strategy for Suppression of Enemy Air Defenses Missions. *Journal of Aerospace Information Systems*, 2013;10(6):287–296. `https://doi.org/10.2514/1.53860`.

[10] Lappas T, Liu K, Terzi E. Finding a Team of Experts in Social Networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09. ACM, New York, NY, USA. 2009 pp. 467–476. ISBN: 978-1-60558-495-9.

[11] Osman N, Gutierrez P, Sierra C. Trustworthy advice. *Knowl.-Based Syst.*, 2015;82:41–59. `https://doi.org/10.1016/j.knosys.2015.02.024`.

[12] Osman N, Sierra C, Sabater-Mir J. Propagation of Opinions in Structural Graphs. In: ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings. 2010 pp. 595–600. doi:10.3233/978-1-60750-606-5-595.

[13] Charlin L, Zemel RS, Boutilier C. A Framework for Optimizing Paper Matching. *CoRR*, 2012. arXiv: 1202.3706 [cs.IR].

[14] Fagin R, Kumar R, Mahdian M, Sivakumar D, Vee E. Comparing and Aggregating Rankings with Ties. In: Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS '04. ACM, New York, NY, USA. 2004 pp. 47–58. ISBN: 158113858X.

[15] Fagin R, Kumar R, Mahdian M, Sivakumar D, Vee E. Comparing partial rankings. *SIAM Journal on Discrete Mathematics*, 2006;20(3):628–648. `https://doi.org/10.1137/05063088X`.

[16] Cormen TH, Stein C, Rivest RL, Leiserson CE. Introduction to Algorithms. McGraw-Hill Higher Education, 2nd edition, 2001. ISBN: 0070131511.

[17] Kamvar SD, Schlosser MT, Garcia-Molina H. The Eigentrust Algorithm for Reputation Management in P2P Networks. In: Proceedings of the 12th International Conference on World Wide Web, WWW '03. ACM, New York, NY, USA. 2003 pp. 640–651. ISBN: 1-58113-680-3.

[18] Hill T, P L. Statistics: Methods and Applications. StatSoft, Inc., 2005. ISBN: 10:1884233597, 13:9781884233593

[19] Sierra C, Debenham JK. Trust and honour in information-based agency. In: 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006 pp. 1225–1232. doi:10.1145/1160633.1160855.

[20] Critchlow DE. Metric Methods for Analyzing Partially Ranked Data. Lecture Notes in Statistics 34. Springer-Verlag New York, 1 edition, 1985. ISBN: 978-0-387-96288-7,978-1-4612-1106-8.