ARM: An Academic Reputation Model

Nardine Osman, Carles Sierra

Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain

Abstract
With open access gaining momentum, open reviews becomes a more persistent issue. Institutional and multidisciplinary open access repositories play a crucial role in knowledge transfer by enabling immediate accessibility to all kinds of research output. However, they still lack the quantitative assessment of the hosted research items that will facilitate the process of selecting the most relevant and distinguished content. This paper addresses this issue by proposing a computational model based on peer reviews for assessing the reputation of researchers and their research work. Although motivated by open reviews, the model is applicable to any review model. It is developed as an overlay service to existing institutional or other repositories. We argue that by relying on peer opinions, we address some of the pitfalls of current approaches for calculating the reputation of authors and papers. We also introduce a much needed feature for review management, and that is calculating the reputation of reviews and reviewers. Last, but not least, we provide a thorough analysis of the possible attacks on our proposed reputation system and the mechanisms that can help counter them.

Keywords: reputation model, academic reputation, algorithms, EigenTrust, attacks and countermeasures

1. Motivation
There has been a strong move towards open access repositories in the last decade or so. Many funding agencies — such as the UK Research Councils, Canadian funding agencies, American funding agencies, the European
Commission, as well as many universities — are promoting open access by requiring the results of their funded projects to be published in open access repositories. It is a way to ensure that the research they fund has the greatest possible research impact. Academics are also very much interested in open access repositories, as this helps them maximise their research impact. In fact, studies have confirmed that open access articles are more likely to be used and cited than those sitting behind subscription barriers [1]. As a result, a growing number of open access repositories are becoming extremely popular in different fields, such as PLoS ONE for Biology or arXiv for Physics.

With open access gaining momentum, open reviews becomes a more persistent issue. Institutional and multidisciplinary open access repositories play a crucial role in knowledge transfer by enabling immediate accessibility to all kinds of research output. However, they still lack the quantitative assessment of the hosted research items that will facilitate the process of selecting the most relevant and distinguished content. Common currently available metrics, such as number of visits and downloads, do not reflect the quality of a research product, which can only be assessed directly by peers offering their expert opinion together with quantitative ratings based on specific criteria. The articles published in the Frontiers book [2] highlight the need for open reviews.

To address this issue we develop a peer review module, the Academic Reputation Model (ARM), as an overlay service to existing institutional or other repositories. Digital research works hosted in repositories using our module can be evaluated by an unlimited number of peers that offer not only a qualitative assessment in the form of text, but also quantitative measures to build the works reputation. Crucially, our peer review module also includes a reviewer reputation system based on the assessment of reviews themselves, both by the community of users and by other peer reviewers. This allows for a sophisticated scaling of the importance of each review on the overall assessment of a research work, based on the reputation of the reviewer.

As a result of calculating the reputation of authors, reviewers, papers, and reviews, by relying on peer opinions, we argue that the model’s main contribution is addressing some of the pitfalls of current approaches for calculating the reputation of authors and papers. It also introduces a much needed feature for review management, and that is calculating the reputation of reviews and reviewers. This is discussed further in the concluding remarks. We also note that although motivated by open reviews, ARM is not restricted to open reviews but is applicable to any review model.
In what follows, we present the ARM reputation model and how it quantifies the reputation of papers, authors, reviewers, and reviews (Section 2). This is followed by some evaluation where we use simulations to evaluate the correctness of the proposed model (Section 3), and some analysis of the possible attacks and the mechanisms that can help counter them (Section 4). We then close with some concluding remarks (Section 5).

2. ARM: the academic reputation model

2.1. Data and notation

To compute reputation values for papers, authors, reviewers, and reviews, we require a Reputation Data Set, which in practice should be extracted from existing paper repositories.

**Definition 1 (Data).** A Reputation data Set is a tuple \( (P, R, E, D, a, o, v) \), where

- \( P = \{ p_i \}_{i \in \mathcal{P}} \) is a set of papers (e.g. DOIs).
- \( R = \{ r_j \}_{j \in \mathcal{R}} \) is a set of researcher names or identifiers (e.g. the ORCID identifier).
- \( E = \{ e_i \}_{i \in \mathcal{E}} \cup \{ \perp \} \) is a totally ordered evaluation space, where \( e_i \in \mathbb{N}\backslash\{0\} \) and \( e_i < e_j \) iff \( i < j \) and \( \perp \) stands for the absence of evaluation. We suggest the range \((0,100]\), although any other range may be used, and the choice of range will not affect the performance.
- \( D = \{ d_k \}_{k \in \mathcal{K}} \) is a set of evaluation dimensions, such as originality, technical soundness, etc.
- \( a : P \to 2^R \) is a function that gives the authors of a paper.
- \( o : R \times P \times D \times Time \to E \), where \( o(r, p, d, t) \in E \) is a function that gives the opinion of a reviewer, as a value in \( E \), on a dimension \( d \) of a paper \( p \) at a given instant of time \( t \).
- \( v : R \times R \times P \times Time \to E \), where \( v(r, r', p, t) = e \) is a function that gives the judgement of researcher \( r \) over the opinion of researcher \( r' \), on paper
Therefore, a judgement is a reviewer’s opinion about another reviewer’s opinion. Note that while opinions about a paper are made with respect to a given dimension in \( D \), judgements are not related to dimensions. We assume a judgement is only made with respect to one dimension, which describes how good the review is in general.

We will not include the dimension (or the criteria being evaluated, such as originality, soundness, etc.) in the equations to simplify the notation. There are no interactions among dimensions so the set of equations apply to each of the dimensions under evaluation.

Also, we will omit the reference to time in all the equations. Time is essential as all measures are dynamic and thus they evolve along time. We will make the simplifying assumption that all opinions and judgements are maintained in time, that is, they are not modified. Including time would not change the essence of the equations, it will simply make the computation complexity heavier.

Finally, if a data set allowed for papers, reviews, and/or judgements to have different versions, then our model simply considers the latest version only.

### 2.2. Reputation of a paper

We say the reputation of a paper \( p \) is a weighted aggregation of its reviews, \( o(r, p) \), where the weight is the reputation of the reviewer, \( R_V(r) \) (Section 2.4).

\[
R_P(p) = \begin{cases} 
\frac{\sum_{r \in rev(p)} R_V(r) \cdot o(r, p)}{\sum_{r \in rev(p)} R_V(r)} & \text{if } \lvert rev(p) \rvert \geq k \\
\bot & \text{otherwise}
\end{cases}
\]

(1)

where \( rev(p) = \{ r \in R \mid o(r, p) \neq \bot \land R_V(r) > \zeta \} \) denotes the reputable reviewers of a given paper. Note that we only consider reviews from reviewers

---

1In tools 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.

---

4
with a reputation larger than a threshold $\zeta$. This guarantees the quality of reviews and helps combat some attacks (as discussed in Section 4). However, as the initial value of a reviewer’s reputation is 0.5, then we require $\zeta \leq 0.5$, to allow the reviews of new researchers to be taken into account.

Note that when a paper receives less than $k$ reviews, its reputation is defined as unknown, or $\bot$. We currently leave $k$ as a parameter, though we suggest that $k > 1$, so that the reputation of a paper is not dependent on a single review. We also recommend small numbers for $k$, such as 2 or 3, because we believe it is usually difficult to obtain reviews. As such, new papers can quickly start building a reputation.

2.3. Reputation of an author

We consider that a researcher’s author reputation is an aggregation of the reputation of her papers. The aggregation is based on the concept that the impact of a paper’s reputation on its authors’ reputation is inversely proportional to the total number of its authors. In other words, if one researcher is the sole author of a paper, then this author is the only person responsible for this paper, and any (positive or negative) feedback about this paper is propagated as is to its sole author. However, if the researcher has co-authored the paper with several other researchers, then the impact (whether positive or negative) that this paper has on the researcher decreases with the increasing number of co-authors. We argue that collaborating with different researchers usually increases the quality of a research work since the combined expertise of more than one researcher is always better than the expertise of a single researcher. Nevertheless, the gain in a researcher’s reputation decreases as the number of co-authors increase. Hence, our model might cause researchers to be more careful when selecting their collaborators, since they should aim at increasing the quality of the papers they produce in such a way that the gain for each author is still larger than the gain it could have received if it was to work on the same research problem on her own. As such, adding authors who do not contribute to the quality of the paper will also discouraged.

\[ R_A(r) = \begin{cases} \frac{\sum_{p \in \text{pap}(r)} \gamma(p)^\alpha \times R_P(p) + (1 - \gamma(p)^\alpha) \times 50}{|\text{pap}(r)|} & \text{if } \text{pap}(r) \neq \emptyset \\ \bot & \text{otherwise} \end{cases} \]  

(2)
where \( \text{pap}(r) = \{ p \in P \mid r \in a(p) \land R_P(p) \neq \bot \} \) denotes the papers authored by a given researcher \( r \), \( \bot \) describes ignorance, \( \gamma(p) = \frac{1}{|a(p)|} \) is the coefficient that takes into consideration the number of authors of a paper (recall that \( a(p) \) denotes the authors of a paper \( p \)), and \( \alpha \) is a tuning factor that controls the rate of decrease of the \( \gamma(p) \) coefficient. Also note the multiplication by 50, which describes ignorance, as 50 is the median of the chosen range \((0, 100]\). If another range was chosen, the median of that range would be used here. The choice of range and its median does not affect the performance of the model (in other words, the results of the simulation of Section 3 would remain the same).

### 2.4. Reputation of a reviewer

Similar to the reputation of authors (Section 2.3), we consider that if a reviewer produces ‘good’ reviews, then the reviewer is considered to be a ‘reputed’ reviewer. Furthermore, we consider that the reputation of a reviewer is essentially an aggregation of the opinions over her reviews.\(^2\)

We assume that the opinions on how good a review is can be obtained, in a first instance, by other reviewers that also reviewed the same paper.\(^3\) However, as this is a new feature to be introduced in open access repositories and conference and journal paper management systems, we believe collecting such information might take some time. An alternative that we consider here is that in the meantime we can use the ‘similarity’ between reviews as a measure of the reviewers opinions about reviews. In other words, the heuristic could be phrased as if my review is similar to yours then I may assume your judgement of my review would be good.

We note \( v^*(r_i, r_j, p) \in E \) for the ‘extended judgement’ of researcher \( r_i \) over researcher \( r_j \)’s opinion on paper \( p \), and define it as an aggregation of

---

\(^2\)We assume a review can only be written by one reviewer, and as such, the number of co-authors of a review is not relevant as it was when calculating the reputation of authors.

\(^3\)To combat potential attacks, we say one can give an opinion about a review only if they already read the paper. And we believe that the only way to ensure someone read a paper is if they left a review about that paper.
opinions and similarities as follows:

\[
v^*(r_i, r_j, p) = \begin{cases} 
  v(r_i, r_j, p) & \text{if } v(r_i, r_j, p) \neq \bot \\
  \text{Sim}(o(r_i, p), o(r_j, p)) & \text{if } v(r_i, r_j, p) = \bot \land o(r_i, p) \neq \bot \land o(r_j, p) \neq \bot \\
  \bot & \text{otherwise}
\end{cases}
\]

(3)

where \(v(r_i, r_j, p)\) represents direct judgements, and \(\text{Sim}\) stands for an appropriate similarity measure. We say the similarity between two opinions is the difference between the two: \(\text{Sim}(o(r_i, p), o(r_j, p)) = 100 - |o(r_i, p) - o(r_j, p)|\).

Given this, we consider that the overall opinion of a researcher on the capacity of another researcher to make good reviews is calculated as follows. Consider the set of judgements of \(r_i\) over reviews made by \(r_j\) as:

\[V^*(r_i, r_j) = \{v^*(r_i, r_j, p) \mid v(r_i, r_j, p) \neq \bot \land p \in P\}\]

This set might be empty. Then, we define the judgement of a reviewer over another one as a simple average:

\[R_V(r_i, r_j) = \begin{cases} 
  \sum_{v \in V^*(r_i, r_j)} v & \text{if } V^*(r_i, r_j) \neq \emptyset \\
  \bot & \text{otherwise}
\end{cases}
\]

(4)

Finally, the reputation of a reviewer \(r_i\), \(R_V(r_i)\), is an aggregation of judgements that her colleagues make about her capability to produce good reviews. We weight this with the reputation of the colleagues as a reviewer:

\[R_V(r_i) = \begin{cases} 
  \sum_{r_i \in R^*(r)} R_V(r_i) \cdot R_V(r_i, r) & \text{if } R^*(r) \neq \emptyset \\
  50 & \text{otherwise}
\end{cases}
\]

(5)

where \(R^*(r) = \{r_i \in R \mid V^*(r_i, r) \neq \emptyset \land R_V(r_i) > \zeta\}\). As with the reputation of a paper (Equation 1), we only consider researchers with a reviewing reputation larger than a threshold \(\zeta\) to guarantee the quality of judgements and help combat some attacks. And again, we require \(\zeta \leq 0.5\) to allow the judgements of new researchers to be considered.

When no judgements have been made over \(r\), we take the value 50 to represent ignorance (as 50 is the median of the chosen range \((0, 100] - \)
again, we note that the choice of range and its median does not affect the performance of the model.

We note here that the reputation of a reviewer depends on the reputation of other reviewers. In other words, every time the reputation of one reviewer will change, it will trigger changing the reputation of other reviewers, which might lead to an infinite loop of modifying the reputation of reviewers. We address this by using an algorithm similar to the EigenTrust algorithm [3]. The similarity to the EigenTrust algorithm is described later on by Section 4.3, and our algorithm is illustrated in detail by Algorithm 10 of the Appendix.

2.5. Reputation of a review

The reputation of a review is similar to the one for papers but using judgements instead of opinions. We say the reputation of a review is a weighted aggregation of its judgements, where the weight is the reputation of the reviewer (Section 2.4).

\[
R_{O}(r', p) = \begin{cases} 
\frac{\sum_{r \in \text{jud}(r', p)} R_V(r) \cdot v^*(r, r', p)}{\sum_{r \in \text{jud}(r', p)} R_V(r)} & \text{if } |\text{jud}(r', p)| \geq k \\
R_V(r') & \text{otherwise}
\end{cases}
\]  
(6)

where \( \text{jud}(r', p) = \{ r \in R \mid v^*(r, r', p) \neq \perp \land R_V(r) > \zeta \} \) denotes the set of reputable judges of a particular review written by \( r' \) on a given paper \( p \). We only consider researchers with a reviewing reputation larger than a threshold \( \zeta \) to guarantee the quality of judgements and help combat some attacks. And again, we require \( \zeta \leq 0.5 \) to allow the judgements of new researchers to be considered.

Note that when a review receives less than \( k \) judgements, its reputation will not depend on the judgements, but it will inherit the reputation of the author of the review (her reputation as a reviewer).

We currently leave \( k \) as a parameter, though as above, we suggest that \( k > 1 \) and we recommend small numbers for \( k \), such as 2 or 3.

2.6. A note on dependencies

Figure 1 shows the dependencies between the different measures (reputation measures, opinions, and judgements). The decision of when to re-
calculate those measures is then based on those dependencies. We provide a summary of this below. Note that measures in white are not calculated, but provided by the users. As such, we only discuss those in grey (grey rectangles represent reputation measures, whereas the grey oval represents the extended judgements).

**Figure 1: Dependencies**

- **Author’s Reputation.** The reputation of an author depends on the reputation of its papers (Equation 2). As such, every time the reputation of one of his papers changes, or every time a new paper is created, the reputation of the author must be recalculated (Algorithm 6 of the Appendix).

- **Paper’s Reputation.** The reputation of a paper depends on the opinions (or reviews) it receives, and the reputation of the reviewers giving those opinions (Equation 1). As such, every time a paper receives a new opinion, or every time the reputation of one of the reviewers changes, then the reputation of the paper must be recalculated (Algorithm 5 of the Appendix).

- **Review’s Reputation.** The reputation of a review depends on the extended judgements it receives, and the reputation of the reviewers giving those judgements (Equation 6). As such, every time a review receives a new extended judgements, or every time the reputation of
one of the reviewers changes, then the reputation of the review must be recalculated (Algorithm 11 of the Appendix).

- **Reviewer’s Reputation.** The reputation of a reviewer depends on the extended judgements of other reviewers and their reputation (Equation 5). As such, the reputation of the reviewer should be modified every time there is a new extended judgement or the reputation of one of the reviewers changes. As the reputation of a reviewer depends on the reputation of reviewers, then we suggest to calculate the reputation of all reviewers repeatedly (in a manner similar to EigenTrust, as illustrated by Section 4.3) until the calculated reputation measures converge (Algorithm 10 of the Appendix). If this will be computationally expensive, then this can be computed once a day, as opposed to triggered by extended judgements and the change in reviewers’ reputation.

- **x-judgement.** The extended judgement is calculated either based on judgements (if available) or the similarity between opinions (when judgements are not available but opinions are) (Equation 3). As such, the extended judgement should be recalculated every time a new (direct) judgement is made, or every time a new opinion is added on a paper which already has opinions by other reviewers (Algorithm 7 of the Appendix).

### 3. Evaluation through simulation

#### 3.1. Simulation

To evaluate the effectiveness of the proposed model, we have simulated a community of researchers, using NetLogo [4]. We clarify that the focus of this work is not implementing a simulation that models the real world, but a simulation that allows us to verify our model. As such, many assumptions that we make for this simulation, and will appear shortly, might not be precisely (or always) true in the real world (such as having the true quality of a paper inherit the quality of the best author).

In our simulation, a breed in NetLogo (or a node in the research community’s graph) represents either a researcher, a paper, a review, or a judgement. The relations between breeds are: (1) *authors of*, that specifies which researchers are authors of a given paper, (2) *reviewers of*, that specifies which researchers are reviewers of a given paper, (3) *reviews of*, that specifies which
reviews give opinions on a given paper, (4) judgments of, that specifies which judgements give opinions on a given review; and (5) judges of, that specifies which researchers have judged which other researcher.

Also, each researcher has four parameters that describe: (1) her reputation as an author, (2) her reputation as a reviewer, (3) her true research quality; and (4) her true reviewing quality. The first two are calculated by our ARM model, and they evolve over time. However, the last two describe the researcher’s true quality with respect to writing papers as well as reviewing papers or other reviews, respectively. In other words, our simulation assumes true qualities exist, and that they are constant over time. In real life, there are no such measures. Furthermore, how good one is at writing papers or writing reviews or making judgements naturally evolves with time. Nevertheless, we chose to keep the simulation simple by sticking to constant true qualities, as the purpose of the simulation is simply to evaluate the correctness of our ARM model.

Similar to researchers, we say each paper has two parameters that describe it: (1) its reputation, which is calculated by our ARM model, and it evolves over time; and (2) its true quality. Again, we assume that a paper’s true quality exists and it is constant over time. How this true quality is calculated is presented shortly.

Reviews also have two parameters: (1) the opinion provided by the review, which in real life is set by the researcher performing the review, while in our simulation it is calculated by the simulator, as illustrated shortly; and (2) the reputation of the review, which is calculated by our ARM model and it evolves over time.

Judgements, on the other hand, only have one parameter: the opinion provided by the judgement, which in real life is set by the researcher judging a review, while in our simulation it is calculated by the simulator, as illustrated shortly.

Simulation starts at time zero with no researchers in the community, and hence, no papers, no reviews, and no judgements. Then, with every tick of the simulation, a new paper is created, which may sometimes require the creation of new researchers (either as authors or reviewers). With the new paper, reviews and judgements are also created. How these elements are created is defined next by the simulator’s parameters and methods, that drive and control its behaviour. We note that a tick of the simulation does not represent a fixed unit in calendar time, but the creation of one single paper.
The ultimate aim of the evaluation is to investigate how close are the calculated reputation values to the true values: the reputation of a researcher as an author, the reputation of a researcher as a reviewer, and the reputation of a paper.

The parameters and methods that drive and control the evolution of the community of researchers and the evolution of their research work are presented below.

1. **Number of authors.** Every time a new paper is created, the simulator assigns authors for this paper. How many authors are assigned is defined by the number of co-authors parameter ($#_{CA}$), which is defined as a Poisson distribution. For every new paper, a random number is generated from this Poisson distribution. Who to assign is chosen randomly from the set of researchers, although sometimes, a new researcher is created and assigned to this paper (see the ‘researchers birth rate’ below). This ensures the number of researchers in the community grows with the number of papers.

2. **Number of reviewers.** Every time a new paper is created, the simulator also assigns reviewers for this paper. How many reviewers are assigned is defined by the number of reviewers parameter ($#_{R}$), which is also defined as a Poisson distribution, with a limit on the minimum number of required reviews ($k$). For every new paper, a random number is generated from this Poisson distribution that respects the minimum number of reviews required. As above, who to assign is chosen randomly from the set of researchers, although sometimes, a new researcher is created and assigned to this paper.

3. **Researchers birth rate.** As illustrated above, every paper requires authors and reviewers to be assigned to it. When assigning authors and reviewers, the simulation will decide whether to assign an already existing researcher (if any) or create a new researcher. This decision is controlled by the researchers birth rate parameter ($birth\_rate$), which specifies the probability of creating a new researcher.

4. **Researcher’s true research quality.** The author’s true quality is sampled from a beta distribution specified by the parameters $\alpha_A$ and $\beta_A$. We choose the beta distribution because it is a very versatile distribution which can be used to model several different shapes of probability distributions by playing with only two parameters, $\alpha$ and $\beta$, as illustrated shortly by our experiments.
5. **Researcher’s true review quality.** The reviewer’s true quality is also sampled from a beta distribution specified by the parameters $\alpha_R$ and $\beta_R$.

6. **Paper’s true quality.** We assume that a paper’s true quality is the true quality of its best author, that is, the author with the highest true research quality. We believe this assumption has some ground in real life. For instance, some behaviour (such as looking for future collaborators, selecting who to give a funding to, etc.) assumes researchers to be of a certain quality, and their research work to follow that quality respectively.

7. **Opinion of a Review.** The opinion presented by a review is specified as the paper’s true quality plus some noise, where the noise depends on the reviewer’s true quality. This noise is chosen randomly from the range $[-(100 - \text{review quality})/2, +(100 - \text{review quality})/2]$. In other words, the maximum noise that can be added for the worst reviewer (whose review quality is 0) is ±50, and the least noise that can be added for the best reviewer (whose review quality is 100) is 0.

8. **Opinion of a Judgement.** The value (or opinion) of a judgement on a review is calculated based on three different inputs: (1) the quality of the judge as a reviewer ($R_V(j)$), (2) the quality, as a reviewer, of the author of the review being judged ($R_V(r)$), and (3) the similarity of the reviews ($\text{sim}$). The equation for calculating a judgement is based on the following intuition. Anyone can spot good reviews, and hence, when the review quality (which is equal to the true quality of the reviewer) is high, then the judgement takes the value of the review quality (or the reviewer quality). However, when reviews aren’t great, then we have two main cases. If the judge has a high reputation as a reviewer, then its judgement depends on the similarity of the reviews. If the reviews are similar, the judgement is high; otherwise, the judgement is low. However, if the judge also does not have a high reputation as a reviewer, then we also have two cases. First, if reviews are similar, then the judgement is high. Otherwise, we will be in the case where neither the judge has a good reputation as a reviewer, nor the author of the review has a good reputation as a reviewer, and the reviews are not similar. In this situation with lots of uncertainty, we say the judgement is closer to ignorance (which we specify as 50, the median of our range $[0, 100]$). The equation that implements this intuition is
defined as follows:

$$\max \{ r_V(r), r_V(j) \cdot \text{sim}, (1-r_V(r)) \cdot \text{sim}, (1-r_V(r)) \cdot (1-r_V(j)) \cdot (1-\text{sim}) \cdot 0.5 \} \cdot 100$$

where $r_V(r) = \frac{R_V(r)}{100}$ and $r_V(j) = \frac{R_V(j)}{100}$. Note that, for simplification, direct judgements have not been simulated, we only rely on indirect judgements.

3.2. Results

3.2.1. Experiment 1: the impact of the community’s quality of reviewers

Given the above, we ran the simulator for 100 ticks (generating 100 papers). We ran the experiment over 6 different cases. In each, we had a number of parameters fixed, which are described by Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required by</th>
</tr>
</thead>
<tbody>
<tr>
<td>$#_{CA}=1$</td>
<td>on average, there are 2 authors per paper</td>
<td>Simulator</td>
</tr>
<tr>
<td>$#_R=2$</td>
<td>on average, there are 2 reviewers per paper</td>
<td>Simulator</td>
</tr>
<tr>
<td>birth-rate=2</td>
<td>the probability of creating a new researcher is 2/100</td>
<td>Simulator</td>
</tr>
<tr>
<td>$\alpha_A=\beta_A=1$</td>
<td>authors quality follows a uniform distribution</td>
<td>Simulator</td>
</tr>
<tr>
<td>$k=2$</td>
<td>minimum # of required reviews/judgements is 2</td>
<td>Simulator &amp; ARM (Equations 1 &amp; 6)</td>
</tr>
<tr>
<td>$\zeta=30$</td>
<td>minimum acceptable researchers’ quality is 30</td>
<td>ARM (Equations 1, 5 &amp; 6)</td>
</tr>
<tr>
<td>$\alpha=1$</td>
<td>tuning factor for co-authors’ impact</td>
<td>ARM (Equation 2)</td>
</tr>
<tr>
<td>$\epsilon=0.10$</td>
<td>EigenTrust convergence parameter</td>
<td>ARM (Algorithm 3)</td>
</tr>
</tbody>
</table>

The only parameters that changed with each case of the simulation where those defining the beta distribution of the reviewers’ qualities, $\alpha_R$ and $\beta_R$. This experiment illustrated the impact of the community’s quality of reviewers on the correctness of the ARM model.

The results are presented by Figure 2, and they show the average of 10 different executions of the simulation (each with 100 ticks, as mentioned
above). For each case, the distribution of the reviewers’ true quality is illustrated to the right of the results. The results, in numbers, are also presented by Table 2. We notice that the least error is presented when the reviewers are all of relatively good quality, with the majority being great reviewers (Figure 2c). The errors start increasing as bad reviewers are added to the community (Figure 2c). They increase even further in both cases, when the quality of reviewers follows a uniform distribution (Figure 2a), as well as when the reviewers are equiprobably good or bad, with no average reviewers (Figure 2b). As soon as the majority of reviewers are of poor quality (Figure 2d), the errors increase even further, with the worst case being when good reviewers are absent from the community (Figure 2f). These results are not surprising. A paper’s true quality is not something that can be measured, or even agreed upon. As such, the trust model depends on the opinions of other researchers. As a result, the better the reviewing quality of researchers, the more accurate the trust model will be, and vice versa.

The results also illustrate how the error in the papers’ reputation increases with the error in the reviewers’ reputation, though at a smaller rate. One curious thing about these results is the constant error in the reputation of authors. The next experiment investigates this issue.

Last, but not least, we note that the error is usually stable. This is because every time a paper is created, all the reviews it receives and the judgements those reviews receive are created at the same simulation time-step. In other words, it is not the case that papers accumulate more reviews and judgements over time, for the error to decrease over time.

Table 2: The results of experiment 1, in numbers

<table>
<thead>
<tr>
<th></th>
<th>Error in Reviewers’ Reputation</th>
<th>Error in Papers’ Reputation</th>
<th>Error in Authors’ Reputation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_R=5$, $\beta_R=1$</td>
<td>$\sim 4.9%$</td>
<td>$\sim 2.3%$</td>
<td>$\sim 15.5%$</td>
</tr>
<tr>
<td>$\alpha_R=2$, $\beta_R=1$</td>
<td>$\sim 11.6%$</td>
<td>$\sim 4.5%$</td>
<td>$\sim 17.7%$</td>
</tr>
<tr>
<td>$\alpha_R=1$, $\beta_R=1$</td>
<td>$\sim 22.0%$</td>
<td>$\sim 6.1%$</td>
<td>$\sim 18.5%$</td>
</tr>
<tr>
<td>$\alpha_R=0.1$, $\beta_R=0.1$</td>
<td>$\sim 32.9%$</td>
<td>$\sim 6.6%$</td>
<td>$\sim 17.7%$</td>
</tr>
<tr>
<td>$\alpha_R=1$, $\beta_R=2$</td>
<td>$\sim 31.7%$</td>
<td>$\sim 9.2%$</td>
<td>$\sim 16.8%$</td>
</tr>
<tr>
<td>$\alpha_R=1$, $\beta_R=5$</td>
<td>$\sim 49.9%$</td>
<td>$\sim 10.5%$</td>
<td>$\sim 17.7%$</td>
</tr>
</tbody>
</table>
distribution of researchers w.r.t. their review quality:

(a) $\alpha_R = 1$ and $\beta_R = 1$

(b) $\alpha_R = 0.1$ and $\beta_R = 0.1$

(c) $\alpha_R = 2$ and $\beta_R = 1$

(d) $\alpha_R = 1$ and $\beta_R = 2$

(e) $\alpha_R = 5$ and $\beta_R = 1$

(f) $\alpha_R = 1$ and $\beta_R = 5$

Figure 2: The impact of reviewers’ quality on reputation measures. For each set of results, the distribution of the reviewers’ true quality is presented to the right of the results.
3.2.2. Experiment 2: the impact of co-authorship

In the second experiment, we investigate the impact of co-authorship on authors’ reputation. We choose the two extreme cases from experiment 1, when there are only relatively good authors in the community ($\alpha = 5$ and $\beta_R = 1$), and when there are only relatively bad authors in the community ($\alpha = 5$ and $\beta_R = 1$). For each of these cases, we then change the number of co-authors, investigating three cases: $\#CA = \{0, 1, 2\}$. All other parameters remain set to those presented in Table 1 above.

The results of this experiment are illustrated by Figure 3, and the results in numbers are presented by Table 3. The results show that the error in the reviewers and papers reputation almost does not change for different numbers of co-authors. However, the error in the reputation of authors does. When there are no co-authors ($\#CA = 0$), the error in authors’ reputation is almost equal to (and even smaller than) the error in papers’ reputation (Figures 3a and 3b). As soon as 1 co-author is added ($\#CA = 0$), the error in authors’ reputation increases (Figures 3c and 3d). When 2 co-authors are added ($\#CA = 2$), the error in authors’ reputation reaches the maximum, around 21-22% (Figures 3e and 3f). In fact, unreported results show that the error in authors’ reputation is almost the same in all cases for $\#CA \geq 2$.

<table>
<thead>
<tr>
<th>$\alpha_R = 5$</th>
<th>$\alpha_R = 1$</th>
<th>$\alpha_R = 5$</th>
<th>$\alpha_R = 1$</th>
<th>$\alpha_R = 5$</th>
<th>$\alpha_R = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta_R = 1$</td>
<td>$\beta_R = 5$</td>
<td>$\beta_R = 1$</td>
<td>$\beta_R = 5$</td>
<td>$\beta_R = 1$</td>
<td>$\beta_R = 5$</td>
</tr>
</tbody>
</table>

| $\#CA = 0$ | $\sim 5.4\%$ | $\sim 49.9\%$ | $\sim 2.4\%$ | $\sim 10.6\%$ | $\sim 1.7\%$ | $\sim 7.9\%$ |
| $\#CA = 1$ | $\sim 4.9\%$ | $\sim 49.9\%$ | $\sim 2.3\%$ | $\sim 10.5\%$ | $\sim 15.5\%$ | $\sim 17.7\%$ |
| $\#CA = 2$ | $\sim 6.4\%$ | $\sim 50.2\%$ | $\sim 2.6\%$ | $\sim 9.9\%$ | $\sim 21.8\%$ | $\sim 21.3\%$ |

4. ARM’s immunity to malicious attacks

In this section, we provide an analysis of the potential attacks on the ARM model and ARM’s immunity (when applicable) to these attacks. An attack is when a peer provides false feedback intentionally. This could happen either to promote/demote one’s or others’ papers as well as promote/demote other researchers. Attacks could also happen simply to ruin the integrity of the
Figure 3: The impact of co-authorship on reputation of authors. For each set of results, the distribution of the reviewers’ true quality is presented to the right of the results.
reputation system. In what follows we describe the attacks against trust and reputation systems in general and analyse to which degree ARM is immune to them.

4.1. Bad mouthing and other false feedback

Bad mouthing is when a peer provides negative feedback on another peer’s performance for the sole purpose of lowering the reputation of that other peer. [5] does not restrict bad mouthing to negative feedback that aims at lowering reputation, but to any feedback that aims at falsely altering reputation. In both cases, the feedback is provided irrespective of the other peer’s performance. Feedback is naturally subjective, and it is hence almost impossible to identify bad mouthing without additional information, such as influencing factors and incentives to bad mouthing.

Giving more weight to feedback coming from reputable peers, however, is one approach that is used to make the reputation algorithm robust to bad mouthing [6]. The ARM model falls under the category of these algorithms that takes into consideration the reputation of the peer providing the feedback. To make the model even more robust to bad mouthing, we additionally place a threshold on the reputation of the researchers providing feedback (reviews and judgements).

Naturally, this assumes that peers with a good reputation provide good feedback, which might not always be the case. For instance, a reputable researcher may occasionally provide maliciously false feedback. This kind of attack is usually referred to as the dynamic personality attack, which we discuss shortly. In other cases, the reputable researcher might simply have made an error, which falls under other false feedback that is discussed next. We note that in both cases, it is extremely difficult to pinpoint such incidents. However, we do note that if there are other honest reputable researchers providing feedback then (1) their feedback could help lessen the impact of the false feedback of the attacker, and (2) we hope that the honest reputable researchers would point out the error in the false feedback, resulting in impacting the reputation of the peer providing false feedback.

We include “nonsense” in our analysis of other false feedback. Nonsense is a form of false feedback that may not necessarily be bad mouthing, in the sense that it may not be applied intentionally. For instance, a school boy entering the system to read papers for his school’s project, and providing nonsense reviews/judgements. Or an expert on anthropology providing a nonsense review/judgement on nano technology. These may be addressed by
(1) verifying the identity of researchers, which is discussed in more detail in Section 4.4, and (2) ensuring that researchers provide feedback only in their areas of expertise. Existing systems based on natural language processing, such as the Toronto paper matching system [7], could be used to learn one’s areas of expertise from his published papers and assess how close his areas of expertise are to a given paper. In such a case, feedback from a researcher \( r \) would only be taken into consideration by the ARM model if researcher \( r \)'s expertise with respect to reviewing a paper \( p \) or judging its review (defined as \( \text{expertise}(r, p) \)) is above a certain acceptable threshold \( \eta \). This requires the definitions of \( \text{rev}(p) \), \( V^*(r_i, r_j) \), and \( \text{jud}(r', p) \) of Equations 1, 4 and 6 to be modified accordingly:

\[
\text{rev}(p) = \{ r \in R | o(r, p) \neq \bot \land R_V(r) > \zeta \land \text{expertise}(r, p) > \eta \} \quad (7)
\]

\[
V^*(r_i, r_j) = \{ v^*(r_i, r_j, p) | v(r_i, r_j, p) \neq \bot \land p \in P \land \text{expertise}(r_i, p) > \eta \} \quad (8)
\]

\[
\text{jud}_{\text{all}}(r', p) = \{ r \in R | v^*(r, r', p) \neq \bot \land R_V(r) > \zeta \land \text{expertise}(r, p) > \eta \} \quad (9)
\]

4.2. Dynamic personality

A dynamic personality attack is when a peer maintains a positive reputation, and yet would abuse this positive reputation every once in a while to act maliciously, as long as this malicious behaviour does not affect its positive reputation. In the academic world, this implies that reputable researchers that have succeeded in building a good reputation for themselves can abuse this reputation to try to promote or demote other papers or friends or foes in their field.

As mentioned earlier, the current ARM model is not fully immune to this attack, but we do note that if there are other honest reputable researchers providing feedback then their feedback could help lessen the impact of the false feedback of the attacker. Additionally, if the honest reputable researchers point out the error in the false feedback, then this will result in impacting the reputation of the attacker over time, that is if s/he continues to abuse their positive reputation.

In the trust and reputation literature the dynamic personality attack has been dealt with by having a memory window so that not all the past history is taken into account. The counter attack becomes even more robust when the memory window is dynamic, where it is shortened when the reputation
is lowered[8]. Future work on the ARM model can further investigate applying solutions such as [8]'s dynamic memory window when calculating a researcher’s reputation as a reviewer.

4.3. Collusion

Collusion happens when a group of peers form a collective in order to collectively subvert the reputation system. For instance, colluders could group to give each other good ratings, or even to give false feedback to someone else.

Detecting and reacting to groups of colluders is an NP-complete problem [9]. Nevertheless, a number of approaches have been proposed in the literature to combat collusion, to some extent. One algorithm that is relevant to our work is the EigenTrust algorithm [3]. EigenTrust is presented by Algorithm 1 below. $C$ is the matrix $[c_{ij}]$, where $c_{ij}$ is peer $i$’s normalised local trust value of peer $j$. The trust vector $\vec{t}$ is the left principal eigenvector of $C$ and its elements, $t_j$, describe the global reputation on $j$. $\vec{e}$ represents the initial global reputation on peers, where all peers have equal reputations. It is an $m$-vector representing a uniform probability distribution over all $m$ peers ($e_i = 1/m$). The basic idea is that the global reputation measures are updated by multiplying the matrix $C$ by the initial global reputation measures. In other words, “the global reputation of each peer $i$ is given by the local trust values assigned to peer $i$ by other peers, weighted by the global reputations of the assigning peers.” [10] This step then is repeated a number of times until the values converge; that is, the difference $\delta$ between the old global reputation measures and the new global reputation measures becomes less than the threshold $\epsilon$. For a detailed explanation of EigenTrust, we refer the interested reader to [3].

**Algorithm 1: EigenTrust**

| $\vec{t}^{(0)} = \vec{e}$; | $\vec{t}^{(0)} = \vec{p}$; |
| repeat | repeat |
| $\vec{t}^{(k+1)} = C^T \vec{t}^{(k)}$; | $\vec{t}^{(k+1)} = C^T \vec{t}^{(k)}$; |
| $\delta = ||\vec{t}^{(k+1)} - \vec{t}^{(k)}||$; | $\delta = ||\vec{t}^{(k+1)} - \vec{t}^{(k)}||$; |
| until $\delta < \epsilon$; | until $\delta < \epsilon$; |
An improved version of the EigenTrust algorithm (Algorithm 2) builds some immunity to collusion by giving higher weight to the feedback coming from pre-trusted peers. Pre-trusted peers are those peers that are known to be trusted. They guarantee the convergence of the algorithm and break up malicious collectives, if they are properly chosen (see [3, Section 4.5] for explanation). As such, it is suggested that pre-trusted peers may be manually hand-picked at the beginning to ensure they do not contain malicious peers.

The improved algorithm is very similar to the previous one. However, in this case, the initial global reputation is the uniform distribution with respect to the pre-trusted peers only. That is, if \( P \) is the set of pre-trusted peers, then the initial global reputation of peers is defined as the vector \( \overrightarrow{p} \), where \( p_i = 1/|P| \) if \( i \in P \), and \( p_i = 0 \) otherwise. The only other difference in the algorithm is that at every iteration, each peer will place some trust on the pre-trusted peers, and the higher the parameter \( a \) is then the higher the confidence on pre-trusted peers (where \( a < 1 \)).

Alternatively to EigenTrust’s approach for combating collusion, social networks may also be used to defend against collusion, again, to some extent. For instance, in the Maze P2P file sharing system [11], the authors make use of a social network and they assume that pre-trusted peers only trust their friends. They then use the friend network of the pre-trusted peers to help detect colluders. In Sorcery [12], peers make use of the overlapping votes of their friends and content providers to decide whether a content provider is a colluder or not. In [13], social distance and interest relationship from a social network is used to identify suspicious collusion.

As illustrated by [13], and despite the rich literature on the subject, neither EigenTrust nor the other proposed algorithms are fully immune to collusion. Nevertheless, we propose future work on ARM to follow EigenTrust’s approach since the ARM algorithm for computing the reputation of reviewers is similar to the EigenTrust algorithm. For instance, our implementation for calculating the reputation of reviewers is similar to EigenTrust, where \( \overrightarrow{t} \) would represent the global reputation of reviewers, whose elements would be the reputation measures \( R_V(r) \), and \( C \) would be the matrix \( [R_V(r_i, r_j)] \) where \( R_V(r_i, r_j) \) describes researcher \( r_i \)’s trust on researcher \( r_j \)’s reviews. However, instead of multiplying \( \overrightarrow{t} \) by \( C \) like EigenTrust, we follow Equation 5. Note that while EigenTrust ensures that the trust measures (in \( C \) and \( \overrightarrow{t} \)) are normalised, ARM does not require trust measures to be normalised. As such, when multiplying \( \overrightarrow{t} \) by \( C \), we require to divide by \( \sum_{r \in R_V} R_V(r) \) to ensure our
reputation measures are within our chosen range (in this case (0, 100]). As such, we replace $CT \to t^{(k)}$ with $\frac{CT \to t^{(k)}}{\sum t^{(k)}}$, resulting in Algorithm 3. Also note that in the ARM model, the initial trust on reviewers is 50 (the median of the selected range (0, 100)). As such, $e^*$ is defined such that $\forall r \cdot e^*(r) = 50$.

To combat collusion, we suggest to follow EigenTrust’s approach. Hence, we suggest to modify our algorithm to follow EigenTrust’s improved version (Algorithm 2), resulting in Algorithm 4, where $\overrightarrow{p}$ is the initial global reputation of reviewers. Unlike EigenTrust, however, we say the initial global reputation of any reviewer should be defined as $p_i = 50(1+\lambda)$, where $\lambda \in [0, 1]$ is used to describe the level of initial trust on a reviewer. If $p_i \notin P$, then we must have $\lambda = 0$, resulting in the initial reputation of $p_i = 50$ that describes ignorance. However, if $p_i \in P$, then we must have $\lambda \in (0, 1]$, resulting in $p_i$’s initial reputation falling in the range (50, 100]. The idea is that not all pre-trusted reviewers may be equal, so $\lambda$ is used to describe how much trusted is the pre-trusted reviewer. The higher $\lambda$ is then the higher the initial reputation is. Finally, we note that selecting pre-trusted peers can be performed by data mining the list of editors and chairs of the top journals and conferences in a given field, and $\lambda$ may even be chosen based on the rankings of such journals and conferences. All of this, however, is left for future work.

<table>
<thead>
<tr>
<th>Algorithm 3: ARM’s reviewers reputation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\overrightarrow{t}^{(0)} = e^*$;</td>
</tr>
<tr>
<td>repeat</td>
</tr>
<tr>
<td>$\overrightarrow{t}^{(k+1)} = \frac{CT \to t^{(k)}}{\sum t^{(k)}}$;</td>
</tr>
<tr>
<td>$\delta =</td>
</tr>
<tr>
<td>until $\delta &lt; \epsilon$;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm 4: Suggested ARM’s improved reviewers reputation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\overrightarrow{t}^{(0)} = \overrightarrow{p}$;</td>
</tr>
<tr>
<td>repeat</td>
</tr>
<tr>
<td>$\overrightarrow{t}^{(k+1)} = \frac{CT \to t^{(k)}}{\sum t^{(k)}}$;</td>
</tr>
<tr>
<td>$\overrightarrow{t}^{(k+1)} = (1 - a) \overrightarrow{t}^{(k+1)} + a \overrightarrow{p}$;</td>
</tr>
<tr>
<td>$\delta =</td>
</tr>
<tr>
<td>until $\delta &lt; \epsilon$;</td>
</tr>
</tbody>
</table>

4.4. Ballot-stuffing

Ballot-stuffing is an attack where a peer provides feedback more often than allowed. In our scenario, this implies a researcher providing more than one review for the same paper, or providing more than one judgement for the same review. Clearly, any system can prohibit this. However, ballot-stuffing
becomes a problem in open systems where it is easy to create false identities. As a result, a peer may abuse such false identities to carry out ballot-stuffing.

We argue that creating false identities in the research world is not an easy matter. First, it is straightforward to verify the identity of a researcher through their email affiliation, which is usually a university email, a research institute email, etc. We naturally assume that institutions are not fraudulent, so they do not generate fake identities. If this is done, which is the case with Google Scholar for instance, it is then easy to categorise verified identities against unverified identities. This could then allow us to detect ballot-stuffing as well as other attacks, such as Sybil, which we discuss shortly. For instance, we currently put a threshold on the reputation of researchers providing a review or a judgement. In addition to this threshold the ARM model could then add a threshold on the number or percentage of feedback (reviews or judgements) coming from unverified accounts. In this case, the definitions of \( \text{rev}(p) \), \( R^*(r) \), and \( \text{jud}(r', p) \) of Equations 1, 5 and 6 are modified accordingly:

\[
\text{rev}(p) = \begin{cases} 
\text{rev}_\text{all}(p) & \text{if } \frac{\text{rev}_\text{ver}(p)}{\text{rev}_\text{all}(p)} > \xi \\
\text{rev}_\text{ver}(p) & \text{otherwise}
\end{cases}
\]  

(10)

where \( \text{rev}_\text{all}(p) = \{ r \in R \mid o(r, p) \neq \perp \land R_V(r) > \zeta \} \) is the set of reputable reviewers of paper \( p \), and \( \text{rev}_\text{ver}(p) = \text{rev}_\text{all}(p) \cap \text{verified identities}(R) \) is the set of reputable reviewers that have verified identities (where \( \text{verified identities}(R) \) represents the set of researchers with verified identities. In other words, if the percentage of reviewers with verified identities is larger than \( \xi \), then the entire set of reviewers is accepted, otherwise, just the set of reviewers with verified identities is considered. We suggest the value of \( \xi \) to be very high (such as 0.95), since we believe the number of unverified identities should be very low in the community.

Similarly, we have:

\[
R^*(r) = \begin{cases} 
R^*_\text{all}(r) & \text{if } \frac{R^*_\text{ver}(r)}{R^*_\text{all}(r)} > \xi \\
R^*_\text{ver}(r) & \text{otherwise}
\end{cases}
\]  

(11)

where \( R^*_\text{all}(r) = \{ r_i \in R \mid V^*(r_i, r) \neq \emptyset \land R_V(r_i) > \zeta \} \) is the set of reputable reviewers that judged researcher \( r \), and \( R^*_\text{ver}(r) = R^*_\text{all}(r) \cap \text{verified identities}(R) \) is the set of reputable reviewers with a verified identity that judged researcher \( r \).
And finally, we have:

\[
jud(r', p) = \begin{cases} 
jud_{all}(r', p) & \text{if } \frac{jud_{ver}(r', p)}{jud_{all}(r', p)} > \xi \\
jud_{ver}(r', p) & \text{otherwise}
\end{cases}
\]

(12)

where \(jud_{all}(r', p) = \{ r \in R \mid v^*(r, r', p) \not= \bot \land R_V(r) > \zeta \} \) is the set of reputable reviewers that judged \( r' \) review of paper \( p \), and \( jud_{ver}(r', p) = jud_{all}(r', p) \cap verified\_identities(R) \) is the set of reputable reviewers with verified identities that judged \( r' \) review of paper \( p \).

As illustrated by Section 5, ARM has already been implemented by two online repositories: DIGITAL.CSIC\(^4\) and e-IEO.\(^5\) ARM will also be used by the ConfMaster conference management system\(^6\) as a trial for the IJCAI 2017 conference.\(^7\) In these cases, the pool of researchers using these systems are real reputable researchers without fake identities. However, we highlight that if reputation is involved, these systems will need to become more strict with their identity verification techniques to close the door on such potential attacks.

4.5. Whitewashing/Newcomer

The whitewashing or newcomer attack is when a peer changes its identity to escape previous bad feedback. As discussed above (Section 4.4), creating false identities in the research world is not an easy matter. We have illustrated above how verifying a researcher’s identity can be (and is being)\(^8\) carried out, and we have noted that we require systems that use the ARM model to closely examine their identity verification techniques.

However, we note that even if a researcher is capable of creating a new identity and starting from scratch, this is not an attractive option for the researcher and this does not really impact the ARM reputation system. Whitewashing is efficient in systems where the initial reputation of a newcomer is sufficiently high enough to benefit the peer changing its identity. In that case, as soon as a peer’s reputation level drops below this initial value, it creates a new identity. However, in the ARM model, newcomers start with an unknown “author reputation” and an average “reviewer reputation”. Hence, if

\(^4\)https://digital.csic.es
\(^5\)http://www.repositorio.ieo.es/e-ieo/
\(^6\)http://www.confmaster.net
\(^7\)http://ijcai-17.org
\(^8\)By Google Scholar, for instance.
the researcher is changing its identity (say switching from one email affiliation to another) for the purpose of abusing its new average reviewer reputation, then the researcher is losing its own author reputation that it has worked for so far and starting from scratch. As author reputation is far more important for one’s career, such an action does not really benefit the researcher. Furthermore, feedback from one average reviewer cannot solely manipulate the reputation of other papers or reviewers. As such, whitewashing intended for abusing the new average reviewer reputation cannot have much impact neither on the researcher nor on the reputation system. Similarly, if the researcher is changing its identity for the purpose of erasing its old negative author reputation, then this does not really impact other researchers and their papers. Furthermore, author reputation is the result of the quality of their research work (i.e. the quality of their papers), and it is usually difficult for one researcher to suddenly change the quality of its research work. One cannot go from being a poor researcher to an excellent researcher overnight. So in the worst case scenario, the newcomer starts as a new researcher in the community, and if their research work can increase their reputation with time, well then good for them!

A note on newcomers. We note that newcomers start off with an average reviewer reputation. And since the ARM model has been designed to rely on the weighted average, the result is that a few new reviewers can easily sway the feedback of one reputable reviewer. Although this is not an attack per se, future work should further analyse and address this issue. In fact, this is not limited to newcomers. Even a few reviewers with poor reputation can sway the feedback of one reputable reviewer. Though the impact is stronger with newcomers since they start off with a higher reputation (an average reputation).

4.6. Sybil

A Sybil attack is an attack where a peer is capable of creating many fake identities, and using these identities to manipulate one’s or other’s reputation through false feedback. This is similar to the collusion attack of Section 4.3;

9In real life scenarios, it is usually difficult to change identities where all these identities get classified as verified accounts. As such, we note that if the author is creating a new identity with an unverified account, then our discussion in Section 4.4 has already illustrated how the impact of unverified accounts can be minimised.
however, instead of relying on colluding peers the attacker uses fake peers with fake identities.

One of the main mechanisms for countering Sybil attacks is through identity verification [5, 6]. We believe the ARM model is resistant to the Sybil attack as creating fake identities is not easy in the academic world, as discussed earlier in Section 4.4. We remind the reader that we require systems that use the ARM model to closely examine their identity verification techniques. And if a Sybil attack does happen, then as illustrated in Section 4.4, categorising researchers’ identities into verified and unverified ones could help minimise the impact of a Sybil attack. This is because we can simply add a threshold on the number or percentage of feedback (reviews or judgements) coming from unverified accounts. See for instance our proposed solution in Equations 10, 11, and 12.

4.7. Attacks and countermeasures at a glance

In what follows we provide a summary of the attacks and our proposed countermeasures to combat these attacks. The summary is presented by Table 4. In some cases, like bad mouthing and whitewashing, ARM is already immune to the attacks. In the case of dealing with other false feedback, and precisely nonsense, we first require that the system verifies researcher’s identities. We also require using Toronto’s paper matching system [7] (or any other similar system) to calculate a researcher’s expertise with respect to reviewing a given paper or judging its review. Given this measure of expertise, we then modify the definitions of \( \text{rev}(p) \), \( V^*(r_i, r_j) \), and \( \text{jud}(r', p) \) following Equations 7, 8 and 9. This ensures that only the feedback (reviews and judgements) of researchers with a minimum level of expertise in the field is accepted. In other cases, like ballot-stuffing and Sybil attacks, we simply require the system to verify researchers’ identities. By having information about one’s identity, a simple modification to the definitions \( \text{rev}(p) \), \( R^*(r) \), and \( \text{jud}(r', p) \) is then suggested following Equations 10, 11, and 12. This combats these attacks by accepting feedback from unverified identities only if the unverified identities were a minority. Concerning the collusion attack, there is no mechanism that provides full immunity to this attack. However, given our similarity to EigenTrust, we suggest to follow EigenTrust’s approach in combating collusion. We propose to apply the suggested solution of Algorithm 4. Last, but not least, concerning the dynamic personality attack, we suggest to adapt [8]’s dynamic memory window and apply it to ARM’s reviewer reputation (Equation 5).
Table 4: Summary of attacks and suggested countermeasures

<table>
<thead>
<tr>
<th>Attack</th>
<th>Countermeasures</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bad mouthing</strong></td>
<td>immune</td>
</tr>
<tr>
<td><strong>Other false feedback (nonsense)</strong></td>
<td>identity verification + Toronto’s system [7] + Equations 7, 8, &amp; 9</td>
</tr>
<tr>
<td><strong>Dynamic personality</strong></td>
<td>adapt [8]’s dynamic memory window and apply it to Equation 5</td>
</tr>
<tr>
<td><strong>Collusion</strong></td>
<td>apply Algorithm 4</td>
</tr>
<tr>
<td><strong>Ballot-stuffing</strong></td>
<td>identity verification + Equations 10, 11, &amp; 12</td>
</tr>
<tr>
<td><strong>Whitewashing / Newcomer</strong></td>
<td>immune</td>
</tr>
<tr>
<td><strong>Sybil</strong></td>
<td>identity verification + Equations 10, 11, &amp; 12</td>
</tr>
</tbody>
</table>

We note that the attacks and countermeasures are the same for any system with any review model, whether it is an open review, a single-blind review, or a double-blind review. The only difference between these systems is the probability of such attacks happening. For instance, bad mouthing, collusion, ballot-stuffing, and Sybil are usually used to bring down the reputation of peers. Naturally, if the identity of peers is hidden (as in the single-blind and double-blind review process), then this is no longer possible. But the same attacks may still be used to increase the reputation of the attackers. In these cases, the same countermeasures hold.

5. Conclusion

We have presented the ARM reputation model for the academic world. ARM helps calculate the reputation of researchers, both as authors and reviewers, and their research work. Additionally, ARM also calculates the reputation of reviews.

Concerning the reputation of authors, the most commonly used reputation measure is currently the h-index [14]. However, the h-index has its flaws. For instance, the h-index can be manipulated through self-citations [15, 16]. A study has also found the h-index as not providing a significantly more accurate measure of impact than the total number of citations [17]. ARM, on the other hand, bases the reputation of authors on the opinions that their papers receive from other members in their academic community. We believe this should be a more accurate approach, though future work should aim at comparing both approaches.

Concerning the reputation of papers, the most common measure currently
used is the total number of citations a paper gets. Again, this measure can easily be manipulated through self-citations. [18] presents an alternative approach based on the propagation of opinions in structural graphs. It allows papers to build reputation either from the direct reviews it receives, or inherit reputation from the place where the paper is published. In fact, a sophisticated propagation model is proposed to allow reputation to propagate upwards as well as downwards in structural graphs (e.g. from a section to a chapter to a book, and vice versa). Although simulations presented in [19] illustrate the potential impact of this model, its drawback is that it remains a theoretical model without any practical implementations since structural information is not currently widely available.

Concerning the reputation of reviews and reviewers, to our knowledge, these reputation measures have not been addressed yet. Nevertheless, we believe these are important measures. Conference management systems are witnessing a massive increase in paper submissions, and in many disciplines, finding good reviewers is becoming a challenging task. Deciding what papers to accept/reject is sometimes a challenge for conference and workshop organisers. ARM is a reputation model that addresses these issues by helping recognise the good reviews/reviewers from the bad.

The obvious next steps for ARM is applying it to a real dataset. In fact, the model is currently being integrated with two Spanish repositories: DIGITAL.CSIC (https://digital.csic.es) and e-IEO (http://www.repositorio.ieo.es/e-ieo/). However, these repositories do not have any opinions or judgements yet, and as such, time is needed to start collecting this data. We are also working with the IJCAI 2017 conference (http://ijcai-17.org) in order to allow reviewers to judge each other. We will use the data from this conference, which will provide us with the reviews and judgements needed for evaluating our model. We will also continue to look through existing datasets.

Future work can investigate a number of additional issues. For instance, we plan to provide data on the convergence performance of the algorithm with respect to calculating the reputation of reviewers, which follows an approach similar to EigenTrust. We also plan to investigate counter measures for the dynamic personality attack and the collusion attack, following the suggestions presented earlier. Additionally, we plan to study alternative mechanisms that would allow us to take into consideration the number of items affecting a reputation measure, such as the number of papers affecting the reputation of an author, the number of reviews affecting the reputation of a paper, or the number of judgements affecting the reputation of a researcher.
The idea is that the larger the number of items affecting a reputation measure is, then the more reliable should this reputation measure be. Of course, one of the main attractiveness of this algorithm is its simplicity, which allowed it to gain attention and to get implemented in a number of real life systems. To consider the number of items affecting a reputation measure implies increasing the complexity of this algorithm — for instance, by working with reputation measures defined as probability distributions as opposed to numerals. Similarly, relying on the weighted average in our calculations have contributed to the simplicity and attractiveness of the ARM model. However, one of the side-effects of using the weighted average is that a few less reputable reviewers (or just new researchers) can easily nullify the feedback of one reputable reviewer. It is debatable whether this feature is desired or not. Nevertheless, alternative approaches may further be investigated that address this issue. Additional future work could also investigate the similarity of reviews. These are now computed based on the similarity of the quantitative opinions, although the similarity between qualitative opinions may also be used in future work by making use of natural language processing techniques. Last, but not least, while we argue that direct opinion can help the model avoid the pitfalls of the literature, it is also true that direct opinions are usually scarce. As such, if needed, other information sources for opinions may also be considered, such as citations. This information can be translated into opinions, and the equations of ARM could then be changed to give more weight to direct opinions than other information sources.

Acknowledgements

This work has been supported by Agreement Computing (a project funded by the Spanish Ministry of Economy & Competitiveness (MINECO), grant # TIN2015-68950-REDC), EVOLVE (a project funded by The Spanish National Research Council (CSIC), grant # 201550E076), CollectiveMind (a project funded by the Spanish Ministry of Economy & Competitiveness (MINECO), grant # TEC2013-49430-EXP), and Open Peer Review Module for Repositories (a project funded by OpenAIRE, which in turn is an EU funded project).

References


URL https://books.google.es/books?id=qJfLBQAAQBAJ

URL http://dx.doi.org/10.1109/CTS.2006.22


URL http://dx.doi.org/10.1109/TC.2012.158

URL http://www.pnas.org/content/102/46/16569.abstract

URL http://dx.doi.org/10.1007/s11192-010-0306-5

URL http://dx.doi.org/10.1002/asi.22976


Appendix A. The ARM Algorithms

Algorithm 5: Reputation of a paper

Function ReputationPaper(p : P) : [0,100] =

Data: p : P /* a paper identifier */
Data: aut : P → R list /* function returning the list of authors of papers */
Data: o : (R × E) list /* list of evaluations of reviewers over paper p */
Data: k : integer /* minimum number of reviewers to compute non-default reputation k > 1 */
Result: RepPaper : [0,100] /* the reputation value of paper p */

/* This function computes the reputation of a paper for a given dimension. It must be called every time a new review is created for this paper, and every time the reputation of one of the paper’s reviewers is modified. */

rev = ∅;
for (r,e) ∈ o do
  if R_R(r) ≠ null then
    rev = rev ∪ (r,e);
  if length(rev) < k then
    RepPaper ← null;
  else
    normal ← 0;
    for (r,e) ∈ rev do
      normal ← normal + ReputationReviewer(r);
    num ← 0.0;
    for (r,e) ∈ rev do
      num ← num + ReputationReviewer(r) * e;
    RepPaper ← num/normal;
return RepPaper;
Algorithm 6: Reputation of an author

Function $ReputationAuthor(r : R)[0, 100]=$

Data: $r : R /* a researcher identifier */$
Data: $pap : R \rightarrow P list /* function returning the list of$
papers of authors */
Data: $aut : P \rightarrow R list /* function returning the list of$
authors of papers */
Data: $alpha : real /* tuning factor for coefficient gamma */$
Result: $RepAuthor : [0, 100] /* the reputation value of$
author $r */$

/* This function computes the reputation of an author. It must be called every time a new paper is created for this author, and every time the reputation of one of the author’s papers is modified. */

$pap2 = \emptyset$;
for $p \in pap(r)$ do
    if $R_P(p) \neq null$ then
        $pap2 = pap2 \cup p$;
    end if
end for
if $pap2 \neq \emptyset$ then
    for $p \in pap2$ do
        $gamma \leftarrow 1/\text{length}(aut(p)) /* length gives the length of a list */$
        $num \leftarrow num + \exp(gamma, alpha)$
        $ReputationPaper(p) + (1 - \exp(gamma, alpha)) \times 50$
        $RepAuthor = num/|pap2|$
    end for
else
    $RepAuthor = null$
end if
return $RepAuthor$
Algorithm 7: Auxiliary functions, used by Algorithms 10 and 11

Function $v^*(r_i : R, r_j : R, p : P) : [0, 100] + \text{null}$

- **Data:** $r_i : R, r_j : R$ /* researcher identifiers */
- **Data:** $p : P$ /* a paper identifier */
- **Data:** $obar : (R \times E^k)\text{ list}$ /* list of vector evaluations of reviewers over $p$ */
- **Data:** $v : (R \times R \times E^k)\text{ list}$ /* list of judgments over paper $p$ */

- **Result:** $\text{extjudge} : [0, 100] + \text{null}$ /* extended judgment of $r_i$ on $r_j$'s opinion of $p$ */

/* This function computes extended judgments. It must be called every time a new judgment is made, and every time a new review is added on a paper which already has reviews by others. It is also called by the AverageJudgment function below and the ReputationReview function of Algorithm 11. */

if $\exists e : (r_i, r_j, e) \in v$ then
  $\text{extjudge} \leftarrow e$
else
  if $\exists ebar, ebar' : (r_i, ebar) \in obar$ and $(r_j, ebar') \in obar$ then
    $\text{extjudge} \leftarrow \text{sim}(ebar, ebar')$
  else
    $\text{extjudge} \leftarrow \text{null}$
return $\text{extjudge}$

Function $\text{sim}(e : E^k, e' : E^k) : [0, 100]$ =

- **Data:** $e : E^k, e' : E^k$ /* evaluation vectors */
- **Result:** $\text{similar} : [0, 100] + \text{null}$ /* difference */

/* This function computes the similarity between two vectors. It is only called by the $v^*$ function above. */

num $\leftarrow 0$;
num' $\leftarrow 0$;
den $\leftarrow 0$;
den' $\leftarrow 0$;
for $i \in [1, k]$ do
  if $e[i] \neq \text{null}$ then
    num $\leftarrow$ num $+$ $e[i]$;
    den $\leftarrow$ den $+$ 1;
  if $e'[i] \neq \text{null}$ then
    num' $\leftarrow$ num' $+$ $e[i]$;
    den' $\leftarrow$ den' $+$ 1;
if den $\neq 0$ and den' $\neq 0$ then
  $x \leftarrow$ num/den;
  $x' \leftarrow$ num'/den';
  $\text{similar} \leftarrow 100 - |x - x'|$
  $\text{similar} \leftarrow \text{null}$;
return $\text{similar}$
Algorithm 8: Auxiliary functions, used by Algorithms 10 and 11 (CONT’D)

Function `AverageJudgment(r : R, r' : R);[0,100]+null =`

Data: `r : R, r' : R` /* two research identifiers */

Result: `AvgJudge : [0,100] + null` /* the average judgment of r over `r'`'s opinions */

/* This function computes the average judgment of one reviewer over another. It is only called by the `ReputationReviewer` function of Algorithm 10. */

`judgements ← 0.0;`
`num ← 0.0;`

for `p ∈ P` do
    if `v*(r, r', p) ≠ null` then
        `judgements ← judgements + 1;`
        `num ← num + v*(r, r', p)`
    if `judgements ≠ 0.0` then
        `AvgJudge ← num/judgements`
    else
        `AvgJudge ← null`

`return AvgJudge`
Algorithm 9: Reputation of a reviewer

Function $ReputationReviewer(r : R) : [0,100] =$

Data: $r : R$ /* a researcher identifier */

Data: $RepReviewer(r) : [0,100]$ /* the reputation value of author $r$ */

Result: $RepReviewerNew(r) : [0,100]$ /* the new reputation value of author $r$ */

/* This function computes the reputation of a single reviewer. It is only called by the function $ReputationReviewers$ and itself, $ReputationReviewer$. */

den ← 0.0;
num ← 0.0;
for $r' \in R, r' \neq r$ do
    if $AverageJudgment(r', r) \neq null$ then
        den ← den + $RepReviewer(r')$;
        num ← num + $RepReviewer(r') \times AverageJudgment(r', r)$;
    if $den > 0.0$ then
        $RepReviewerNew(r)$ ← num/den;
    else
        $RepReviewerNew(r)$ ← 50;
return $RepReviewerNew(r)$;
Algorithm 10: Reputation of a reviewer (CONT’D)

Function ReputationReviewers : [0,100] list=

Data: ϵ : [0,100] /* a threshold specifying when is the difference in reputation considered negligible */
Data: r : R /* a researcher identifier */
Data: RepReviewer(r) : [0,100] /* the reputation of author r ∈ R. Initially it is set to RepReviewer(r) = 50 */
Data: RepReviewer{NEW}(r) : [0,100] /* the reputation value of an author r in R. Used to hold the new value during convergence. */
Result: RepReviewers : [0,100] list /* returns the list of updated reputation value for all authors r in R; that is, RepReviewers = {RepReviewer(r)}∀r∈R */

/* This function computes the reputation of all reviewers. It must be called every time an extended judgment over an opinion of r is created or modified (calculated by the function v∗ of Algorithm 7). Alternatively, this might be called once a day. */

repeat ←− true;
RepReviewersOLD ←− {RepReviewer(r)∀r∈R};
RepReviewersNEW ←− {};
while repeat ≠ false do
   for r ∈ R do
      /* First we calculate the new reputation values, using the old ones */
      RepReviewer{NEW}(r) ← ReputationReviewers(r);
      RepReviewersNEW ←− RepReviewersNEW ∪ RepReviewer{NEW}(r);
   for r ∈ R do
      /* Second we save the newly calculated reputation values as the current ones */
      RepReviewer(r) ← RepReviewer{NEW}(r);
      /* Finally we calculate the vector distance between the old and the new reputation values, that is
      \[ \sqrt{\sum_r (\text{RepReviewer}_{NEW}(r) - \text{RepReviewer}_{OLD}(r))^2} \] */
      if ||RepReviewersOLD − RepReviewersNEW|| ≤ ϵ then
         repeat ←− false;
   return RepReviewersNEW;

39
Algorithm 11: Reputation of a review

Function ReputationReview(r : R, p : P, k : integer) : [0,100] =

Data: r : R /* a researcher identifier */
Data: p : P /* a paper identifier */
Data: k : integer /* minimum number of judgments to compute non-default reputation review value, k > 0 */
Result: RepReview : [0,100] /* the reputation value of the review of r over p */

/* This function computes the reputation of a particular review. It must be called every time an extended judgment over that opinion of r is created of modified (calculated by the function $v^*$ of Algorithm 7), and every time the reputation of the author of the review is modified. */

jud = ∅;
for r' ∈ R, r' ≠ r do
    if $v^*(r', r, p) \neq \text{null} \land R_R(r') \neq \text{null}$ then
        jud = jud ∪ r';
    den ←− 0.0;
    num ←− 0.0;
    if jud ≠ ∅ then
        for r' ∈ jud do
            den ←− den + ReputationReviewer(r');
            num ←− num + ReputationReviewer(r') * $v^*(r', r, p)$
        RepReview ←− num/den;
    else
        RepReview ←− ReputationReviewer(r)
return RepReview;