Dependable Filtering: Philosophy and Realisations

MATTEO DELL'AMICO
Eurecom
and
LICIA CAPRA
University College London

Digital content production and distribution has radically changed our business models. An unprecedented volume of supply is now on offer, whetted by the demand of millions of users from all over the world. Since users cannot be expected to browse through millions of different items to find what they might like, filtering has become a popular technique to connect supply and demand: *trusted* users are first identified, and their opinions are then used to create recommendations. In this domain, users' trustworthiness has been measured according to one of the following two criteria: *taste similarity* (i.e., "I trust those who agree with me"), or *social ties* (i.e., "I trust my friends, and the people that my friends trust"). The former criterion aims at identifying *concordant* users, but is subject to abuse by malicious behaviours. The latter aims at detecting *well-intentioned* users, but fails to capture the natural subjectivity of tastes. In this paper, we propose a new definition of *trusted recommenders*, addressing those users that are *both* well-intentioned and concordant. Based on this characterisation, we propose a novel approach to information filtering that we call *dependable filtering*. We describe alternative algorithms realising this approach, and demonstrate, by means of extensive performance evaluation on a variety of real large-scale datasets, the high degree of both accuracy and robustness they entail.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms: Algorithms, Security, Measurement

Additional Key Words and Phrases: Collaborative filtering, link analysis, profile injection, social networks, Sybil attack

1. INTRODUCTION

In his bestseller "The Long Tail", Chris Anderson [2006] emphasizes how digital distribution has dramatically changed retailers' business models. Traditional retailers have a limited space they can use to stock items; market forces drive them to carry only a limited number of items, in particular, those that have the best chance to sell, thus losing less popular ones. With the advent of the Internet, retailers are

Authors' addresses: Matteo Dell'Amico, Eurecom, 2229 Route des Crêtes, BP 193 F-06560 Sophia-Antipolis cedex, France, matteo.dell-amico@eurecom.fr; Licia Capra, Department of Computer Science, University College London, London WC1E 6BT, United Kingdom, l.capra@cs.ucl.ac.uk.
A preliminary version of this paper appeared in the "Joint iTrust and PST Conferences on Privacy, Trust management and Security" (IFIPTM 2008).

not bound by the same physical constraints, so that a much wider variety of items can be offered from the 'long tail'. As a result, while a traditional bookshop can hardly be expected to sell more than 100,000 different titles, an online service such as Amazon.com can offer its costumers millions of different products. The difference becomes even sharper now that producing the content itself is within easy reach of almost anybody: consider the difference of choice between traditional broadcast or cable TV and sites like YouTube.com. However, as Anderson points out, providing people with a massive choice is detrimental, if that means they have to browse through thousands, or even millions, of potentially relevant items. Rather, people must be assisted in finding what they want. Filters can be used to *connect supply and demand*, making it easier for users to find the particular content that they would enjoy.

The most popular technique to realise this connection is collaborative filtering (CF) [Herlocker et al. 1999]. Most of the work on collaborative filtering has been focusing on identifying users with similar preferences, and then recommending items that people with similar tastes have approved. Traditional collaborative filtering techniques have worked quite well for the mass market and under the assumption of collaborative behaviours. However, these techniques have been subject to abuse by malicious behaviours [Lam et al. 2006]: for example, malicious users could copy honest users' reviews, to gain high similarity scores with them; they could subsequently inject inflated reviews in the system, to trick those users into buying an item or, vice versa, to disrupt an item's sales.

We argue that *accurate* and *robust* filtering techniques can be devised by exploiting information from a user's social network. Throughout this paper, we will refer to this approach as *dependable filtering*. The core idea is to give higher weight to recommendations received from *trusted* users. To be trusted, a user must be both *well intentioned* and *concordant*. Traditional collaborative filtering techniques focus only on concordance (i.e., the ability to give useful - in a subjective way - recommendations), without considering the fact that concordant users may indeed be malicious. Rather than relying on all recommendations from similar (i.e., concordant) users, our approach specifically looks for well-intentioned users (i.e., users who are willing to provide honest recommendations) among those with whom we have stronger social relationships. Social ties are a warranty against malicious behaviors: if the trust inference algorithm is robust, it would be very costly for an attacker to build enough friendships with 'honest' users to effectively subvert the system.

The robustness of CF systems has usually been measured with respect to the proportion of malicious nodes in the network, under the assumptions that attackers are not able to create unlimited new identities at will, and they are not aware of the judgements expressed by others [O'Mahony et al. 2002; Burke et al. 2005; Mobasher et al. 2006; Mobasher et al. 2007]. In our approach, both assumptions are relaxed instead, and the impact of an attack becomes limited only by the connectivity of malicious nodes in the social network (i.e., how many honest users they have fooled into explicitly trusting them as honest).

The remainder of the paper is structured as follows: we begin by analysing the current state of the art in the area of robust recommender systems (Section 2). We

continue by presenting in Section 3 the two main concepts underpinning dependable filtering, that is, intent and concordance. In Section 4 we discuss various alternative realisations of dependable filtering. Section 5 describes the attacks against which dependable filtering must defend itself. Section 6 illustrates how experiments have been unfolded, what datasets have been used (namely, CiteSeer and Last.fm), and what metrics have been analysed (i.e., accuracy and robustness). In Section 7 we report on the results of an extensive simulation study, and in Section 8 we conclude the paper by summarising its contributions and discussing future directions of research. Note that a preliminary version of this work introduced the dependable filtering philosophy and discussed an initial implementation [Dell'Amico and Capra 2008]; this paper extends upon that, by discussing (Section 4) and extensively evaluating (Section 7) three further realisations, as well as better positioning the approach in the literature (Section 2).

2. RELATED WORK

The attack known as "shilling" (or "profile injection") is a well known attack in the recommender systems community [Lam and Riedl 2004; Mobasher et al. 2006]: a group of users, be them humans or agents, enters false opinions in the system, to push for (or to nuke) an item. The unfolding of such an attack in a collaborative filtering setting is quite simple yet powerful: malicious users first enter opinions in the system similar to those of honest users, with the aim to trick the CF system into considering them good recommenders; they then start to falsely praise a new item, which the CF system then recommends to the honest users, by way of their past 'similar' opinions.

Profile injection poses a serious threat to both end-users and businesses that rely on recommender systems. The research community has thus been very active in trying to devise methods that are resilient against this type of attack. In Section 2.1 we review the state of the art in the area of robust recommender systems; while quite sophisticated and highly effective in some scenarios, all such approaches make a common assumption: that is, the number of shills is limited (or, in other words, it is costly for an attacker to create a very high number of shills).

The research problem we address in this paper covers the case in which such an assumption no longer holds: in online environments, where new identities can be created with minimal cost, how can we protect a CF system against attackers who can create an unlimited number of shills? While this problem has been left largely unexplored in recommender systems, it has been touched upon in the area of peer-to-peer (P2P) systems. Section 2.2 draws a comparison between profile injection in collaborative filtering, and Sybil attacks in P2P networks, and illustrates how such attacks can be dealt with by means of social networks, and, in particular, web-of-trust, in such scenario. With the advent of Web 2.0 and its social networking sites, many recommender systems have been enriched with social networking capabilities and could thus leverage such techniques to add resilience to Sybil (shilling) attacks; however, as we review in Section 2.3, none of these solutions has been developed with robustness in mind. In this paper we aim to fill this gap, by proposing and evaluating an approach to robust collaborative filtering, which we call *dependable filtering*, whose main characteristic is the exploitation of social ties to bound the

impact of attacks made by a potentially *unlimited* number of Sybils.

2.1 Profile Injection in Collaborative Filtering

Since shilling attacks were first illustrated [Lam and Riedl 2004], a variety of techniques have been proposed to defend against them. The underpinning idea to most approaches is that, as malicious profiles are synthetic, one could devise algorithms that reveal such fake rating patterns, in order to then isolate them.

For example, O'Mahony et al. [2004] propose a neighbourhood filtering mechanism to isolate false profiles from users' neighbourhoods. Profiles are deemed false if they exhibit features not commonly present in honest user profiles; however, this type of protection can be easily defeated by slightly more sophisticated attackers who know that such countermeasures are in place, and thus adopt less detectable attack patterns. Zhang et al. [2006] propose another technique based on anomaly detection, looking at the growth of a user's profile over a short period of time, to detect attacks early on and thus limit their impact; however, if malicious users dilute their strategy over a longer period of time, the attack would pass unnoticed.

Statistical anomaly detection techniques have been widely studied to reveal rating patterns of shilling attackers. Chirita et al. [2005], for example, propose an effective detection scheme that requires no knowledge about honest users' behaviour (i.e., zero-knowledge attack). Bhaumik et al. [2006] devise an attack detection scheme that aims to identify what items may be under attack, based on rating activity related to the item (rather than by whom). In [Sandvig et al. 2007b; Bhaumik et al. 2007; Mobasher et al. 2007], algorithms based on a data mining technique called association rule mining are proposed and demonstrated to be less susceptible to attacks than traditional recommender system techniques (e.g., k-Nearest Neighbour), while not compromising accuracy. Williams et al. [2007] propose a supervised classification learning technique approach that identifies characteristics of users' profiles that malicious users could exploit to engineer the influence of their attack to the collaborative system. Based on these characteristics, a classifier is then built to distinguish attack profiles from genuine ones. Most of these approaches assume that the filler items in an attack profile have been chosen at random; however, if the target item rating distribution is known (an assumption that is anyway disputable), the filler items could be chosen intelligently, thus seriously magnifying the impact of the attack [Ray and Mahanti 2008]. A variety of other model-based techniques (e.g., clustering, feature reduction, etc.) have been designed to deal with the scaling problem caused by the analysis of very large datasets; a beneficial side effect of these techniques is that the impact of profile injection attacks is weakened ([Sandvig et al. 2008; Mehta and Hofmann 2008]). Common to all the above approaches is the assumption that a bounded number of fake profiles is present in the system at any time; the robustness of all these techniques has thus been measured in terms of number of attackers in place. Such an assumption is valid if it is costly for an attacker to create an unlimited number of profiles; for example, this is the case for systems that allow users to rate items only after purchase [Hurley et al. 2007]. The problem we tackle in this paper is somehow orthogonal: rather than measuring the strength of an attack while varying the percentage of shills in the system, we provide robustness even against an unlimited number of fake users.

The above approaches mainly rely on metrics of profile's similarity to elect neigh-

bours. Resnick and Sami [2007; 2008] proposed the notion of users' *reputation* to elect neighbours instead. Their algorithm, called 'influence limiter', quantifies users' reputation based on their past rating history, and uses it to limit their influence in recommendations. At the cost of discarding useful data from genuine raters, and assuming that the number of injected profiles is below a certain threshold, the damage done by shills gets limited. Similarly, O'Donovan and Smyth [2006] proposed the notion of neighbour's *trust*, computed based on the historical accuracy of the recommendations given by that neighbour; this definition of trust is thus heavily based on similarity, but with an added notion of 'stability' over time. This idea was later expanded by Sandvig et al. [2007a], selecting neighbours based on a more general definition of 'neighbour utility', encompassing both the so called 'trust-based weighting' (computed as above), and a 'significance weighting', computed as size of profile overlap, to prevent neighbours with only a few commonly rated items from dominating predictions. Although the attack model against which such techniques have been assessed remains the same (i.e., a bounded number of fake profiles), improvements in both accuracy and robustness, with respect to traditional recommender systems, have been reported. When presenting our model (Section 3), we will return to the notion of *trusted* neighbours, but our definition of trust will extend beyond the boundaries of similarity (i.e., the information available from the user-rating matrix), to include information coming from the users' social network too.

2.2 Sybil Attacks in Peer-to-Peer Systems

A peer-to-peer (P2P) system [Schollmeier 2001] is a distributed network architecture whereby participants (also called peers or entities) make a portion of their resources (such as CPU, disk storage and network bandwidth) available to other participants, without the need for central coordination. The functioning of a P2P system often relies on a companion reputation system, whose goal is to provide incentives for participants to offer, rather than simply consume, resources [Gupta et al. 2003]. The most common attack to the reputation system of a P2P network is the well-known 'Sybil attack' [Douceur 2002]: a malicious entity creates a large number of pseudonymous entities (Sybils), and uses them to gain a disproportionately large influence in the network. The cheaper the creation of fake pseudonyms, the higher the system's vulnerability.

Sybil attacks have striking similarities to shilling attacks, so one may wonder how the P2P research community has addressed the problem. Many approaches in this area rely on the use of social networks, and, in particular, web-of-trust (WoT). For example, Cheng and Friedman [2005] build a WoT where nodes are entities and edges represent 'direct trust', that is, the outcome of actual interactions. In order to assess the reputation of a given node in the WoT, trust is propagated along paths by means of asymmetric flow functions which are shown to be Sybil-proof. Similarly, Yu et al. [2006] create a WoT whereby edges between nodes indicate human-established trust relationship; while malicious users can create a disproportionally large number of Sybils, it is very costly from them to create trust relationships. The WoT thus exhibits an extremely small "cut" in the graph between the Sybil nodes and the honest nodes. A technique called SybilGuard is then proposed that exploits this property to bound the effectiveness of Sybil

attacks.

Following the popularity of Web 2.0, an increasing number of businesses are enriching their online recommender systems with social networking functionalities [Golbeck 2008], so one may wonder whether Sybil-proof techniques developed for P2P systems could be successfully exploited to defend collaborative filtering too, against a potentially unlimited number of Sybils. We explore this next.

2.3 Social Networks in Collaborative Filtering

Collaborative filtering approaches compute recommendations for a given user by first identifying those neighbours with a similar profile, then recommending items from such profiles. Recently, user studies have highlighted how recommendations received from similar users whom the target user *knows* are preferred over those received from anonymous users, despite their high profile similarity [Bonhard and Sasse 2006]. Social networks have thus been investigated as an alternative to the computation of taste similarity in cases where homophily (the tendency to associate with similar others) is high. For example, Golbeck [2005] proposes the use of trust and social networks in a movie ratings website, and demonstrates that, for users with opinions that are divergent from the average, the trust-based recommended ratings are more accurate than those computed with common collaborative filtering techniques. Similarly, Groh and Ehmig [2007] exploit the manually elicited friends in a Munich-based social-network to predict users' tastes (in this case, related to night clubs), once again demonstrating higher degrees of accuracy than when using traditional CF approaches. Avesani et al. [2005] present a recommender system for ski mountaineers, whereby users not only express opinions about items, but also about how much they trust each other's opinions; in order to be part of a user's neighbourhood, a node in the WoT must be reachable from the target node via a trust propagation algorithm. Massa and Avesani [2007] compare a variety of global and local trust metrics against the Epinions.com community, demonstrating how a local trust metric achieves higher accuracy than a global one in predicting how much trust users should place in each other. Similarly, Zheng et al. [2007] perform a study on a dataset from Amazon.com to assess the impact of incorporating social-network information into CF, with the conclusion that neighbours' selection based on social-network information yielded remarkably more accurate results than those obtained with non-socially-connected neighbours.

All these approaches have been developed with the aim of improving the accuracy of collaborative filtering. None of them has looked at the problem of robustness; indeed, none of the trust propagation schemes they adopt is resilient to Sybil attacks. Note that avoiding trust propagation and relying solely on the recommendations coming from the hand-picked trusted recommenders each user elicit is not an option, as an extremely low number of items would become recommendable, causing the recommender system to loose its appeal. The stance we take is thus different: we exploit social networks primarily to gain robustness against Sybil attacks; as we shall demonstrate during our experimental evaluation (Section 7.1), homophily is also intrinsically exploited (if present) to boost accuracy, although we rely on taste similarity as our main means to guarantee accuracy. We note that other researchers have started to look at social networks as a primary means to protect against attacks: Benevenuto et al. [2008], for example, have used a machine learn-

ing algorithm to detect video spammers in the YouTube.com website, and found that the algorithm works best (i.e., it detects spammers with the highest degree of accuracy) when trained on social network data. In other words, social network attributes are the best discriminator between legitimate users and spammers (as opposed to users' profiles and video attributes).

3. DEPENDABLE FILTERING - PHILOSOPHY

Dependable filtering relies on the identification of *trusted* recommenders. In the scope of this work, in order to be trusted, a recommender must be both *well-intentioned* and *concordant*. The three questions we are thus trying to answer are: (1) how to evaluate intention (Section 3.1); (2) how to evaluate concordance (Section 3.2); and (3) how to combine this information to find trusted recommenders (Section 3.3).

3.1 Intent - Trust over Users

We define intent as the *willingness* of a user to provide honest judgements, differentiating "spammers" from people who are legitimately using the system. In this paper, we use the general term 'judgements' to reflect the fact that our approach is equally applicable to 'endorsements' of products or content, as to 'negative' or purely 'informative' statements (e.g., "avoid that restaurant" or "this is relaxing music"). The well studied case of numeric rating (e.g., "this movie is worth 3 stars out of 5") can be seen as a more specific instance of this general framework. Note that a judgement given with good intent should not necessarily be recommended, since users may have different tastes and preferences; the next section will illustrate how to find concordant users among well-intentioned ones, in order to select what judgements to recommend.

In our approach, a well-intentioned user is one that has a high *reputation*. We use the word 'reputation' here in its most general sense, that is, 'the estimation in which a person or object is held by the community or public' (source: Oxford Dictionary). Reputation is built over time: the more cooperative a user has been in the past, the higher their reputation. Note that high reputation is not developed by intrinsically honest users only: a selfish individual can be motivated to cooperate by sufficient incentives. If users benefit enough from being cooperative, reciprocative patterns emerge, and selfish individuals will maximize their payoff by cooperating [Axelrod 1984; Nowak and Sigmund 1998].

Finding users with high reputation is rather straightforward in centrally-managed systems: the system acts as a trusted third party, witnessing interactions and recording outcomes; if Alice (A) wishes to find out the reputation of Bob (B), A can simply query the system to get the answer she is looking for. Finding nodes with high reputation in distributed environments is not that trivial: interactions are not witnessed by trusted third parties, so it is not possible to keep global reputation records, as malicious users could report false outcomes of their interactions with other peers, in order to damage their reputation.

How can we find well-intentioned users in this case? Instead of global reputation, reciprocal *trust* relationships between nodes are maintained by the nodes themselves in the form of a *web of trust*, that is, a directed graph where nodes are users and an edge from A to B indicates that A trusts B (i.e., A has had direct interactions

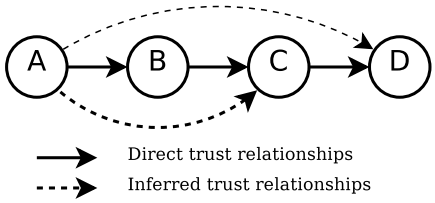


Fig. 1: Transitive trust propagation pattern.

with B in the past and has reported she trusts B). A 'web of trust' is thus an instance of a social network where links represent assessments on the behaviour of nodes rather than simple acquaintance.

The web of trust can be built both explicitely and implicitely: the former requires users to manualy create their social network, for example, by means of a functionality similar to the "Add as a friend" one available on sites like MySpace or FaceBook. The latter builds the user's social network automatically instead, for example, by monitoring the user's email activity, by extracting phone-book contacts, or as described in ReferralWeb [Kautz et al. 1997]. We are not concerned with the specific technique used to create the web of trust, and both implicit and explicit techniques are viable, as long as it is costly, for malicious users, to obtain endorsements from honest ones. For this reason, we do discourage approaches that automatically create the web of trust purely from users' similarity: this is because, as we shall further discuss in the next section, the fact that two users have expressed the same judgements says nothing about their intent.

A problem arises when A has to judge the intentions of C, with whom she has never interacted before. In this case, it is sensible to give some trust to nodes that are recommended by nodes that we trust (and, iteratively, to nodes trusted by these new nodes as well). In other words, *A propagates trust over intent*, from A itself to all nodes reachable from it via a directed path over the web of trust. It does so by means of the so called *transitive trust* propagation pattern (Fig. 1). Usually, the level of trust inferred over a path is lower or equal to the minimum value of trust over the path; moreover, trust is generally dispersed along longer paths, meaning that a "friend of a friend" is usually trusted less than a friend, and so on. Although these properties are recurrent, the details differ between the different instances of algorithms, as we will more concretely see in Section 4.1.

The principle of trust transitivity has been criticized since the judgement of who deserves trust is subjective [Langheinrich 2003; Jösang 1996] (i.e., we are not guaranteed to like all the friends of our friends). However, we argue that, unlike concordance (discussed in the next section), benevolent intent is a concept where subjectivity does not apply strongly: if node C has given honest judgements to node B (i.e., B trusts C), it is likely that C will do the same with node A too (A will trust C), as reciprocative behaviour creates an incentive for node C to be consistently well-behaved, thus building a high reputation [Feldman et al. 2004].

3.2 Concordance - Trust Over Judgements

Reasoning about intent intuitively helps building *robust* filtering techniques; however, intent alone does not capture the high degree of subjectivity inherent in tastes.

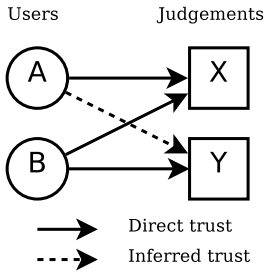


Fig. 2: Co-citation trust propagation pattern.

In order to build *accurate* filters, we thus enrich our definition of trusted recommenders with concordance. In this work, we define *concordant* those users who have expressed the same judgement; since judgments are inherently subjective, concordance is a subjective matter as well.

A sensible way of evaluating concordance is via the so called *co-citation pattern*. A bipartite graph is used to represent a *network of judgments*: users (e.g., $\{A, B\}$) and judgments (e.g., $\{X, Y\}$) form two disjoint sets of vertexes; an edge (A, X) is present if user A expressed the judgment X. If users A and B agree on judgment X (i.e., there exist edges $A \to X$ and $B \to X$), then A may consider B a concordant user. Using the co-citation pattern (Fig. 2), she may then *propagate trust over concordance* on the other judgements that B expressed.

In the previous section, we argued that users' intent is not sufficient to warrant trust to their judgements; the same can be said for concordant users. Let us consider, for instance, a malicious user Mallory, wishing to trick Alice in believing a dishonest judgement Z stating that "Mallory's Greasy Restaurant offers very good food". In order to do so, Mallory could simply copy Alice's judgements; using the co-citation trust propagation pattern, Alice would deem Mallory a very concordant evaluator, and would consequently believe/trust judgement Z too.

We thus call for the combination of intent and concordance in order to identify *trustworthy recommenders*, that is, recommenders who are willing to provide us with honest judgements and that we are likely to agree with.

3.3 The Combined Approach

As discussed above, using the *transitivity* trust propagation pattern alone is not enough, as subjectivity of tastes, which is an intrinsic characteristic of judgments, is lost. On the other hand, using the *co-citation* trust propagation pattern alone is subject to abuse by malicious users.

We propose a novel approach that combines the strengths of the two patterns, while circumventing their individual weaknesses: we exploit the transitivity trust propagation pattern on the web of trust to determine well intentioned users, and the co-citation trust propagation pattern on the network of judgments to evaluate their concordance. By so doing, we are capable of inferring trust over judgments, in a way that is both accurate and robust. The underpinning idea is that, in order to be trusted, a judgment must have been expressed by a user who is both *willing* (intent) and *able* (concordance) to give useful judgments. We call the new approach

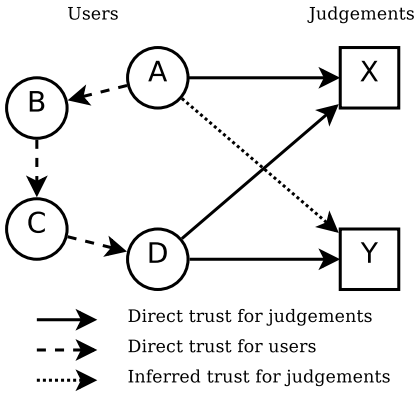


Fig. 3: Combined trust-propagation approach.

dependable filtering. Based on the interpretation of trust propagation over intent and concordance we gave in the previous two sections, A can infer trust for a judgment Y expressed by a user D (Fig. 3) if:

(1) there exists a directed path from A to D in the web of trust (e.g., $A \to B \to C \to D$);
(2) A and D both expressed at least one common judgment (e.g., X).

This is the first approach we are aware of that aims at increasing the utility of recommendations, by exploiting information coming from the social network *and* from individual's preferences at the same time. We are aware of only two other works where the transitivity and co-citation trust propagation patterns have been used together, but with rather different goals and following a different philosophy: Guha et al. [2004] propagate trust using *either* co-citation or transitivity in a social network where links represent similarity in preferences; Massa and Avesani [2007], use the transitive trust propagation pattern as an *alternative to* the co-citation pattern, in order to bootstrap trust when traditional user similarity cannot be computed, again because of lack of information. As already discussed in Section 2.3, these approaches work well in those scenarios where there is a strong correlation between social ties and individual preferences (homophily). On the contrary, our approach is best suited in those scenarios where the social network is not just a surrogate of users' preferences. As we shall demonstrate in Section 7, when separate information is available about the web of trust and judgments, an approach that reasons about intent and concordance *at the same time* can yield the biggest increase in the utility of recommendations, even in the absence of malign behavior. Before moving on to evaluating the approach, we present various algorithms that can be combined to offer alternative realisations of dependale filtering.

4. DEPENDABLE FILTERING - REALISATIONS

In the previous section, we have introduced dependable filtering from a philosophical viewpoint, highlighting the advantages of propagating trust over both intent and concordance, in order to give users *trusted judgements*. To be of practical use, an implementation of dependable filtering would need to attribute a numeric value to

the *amount* of trust a judgement deserves. This would ultimately allow users to rank judgements and/or to filter out unreliable ones.

In this section, we first review the literature to discuss algorithms that implement the two steps of dependable filtering (i.e., transitive trust propagation pattern and co-citation trust propagation pattern) separately. We then focus on particular combinations of these algorithms that realise the whole dependable filtering approach and that will later be the focus of our evaluation.

In describing the various algorithms, we will refer to the general case of weighted social networks, with weights expressing the strength of social ties. The user-judgement edges can be weighted as well, representing the level of confidence of a user towards a given judgement. The unweighted case is just a specific instance of the more general one, with all instances of trust relationships and/or judgements having the same weight.

4.1 How to Evaluate Intent

The first part of our approach requires a navigation of the web of trust, in order to compute intent values for all users, thus isolating those users that are deemed malicious. Various algorithms exist to quantify the amount of trust that is propagated transitively on a weighted social network. Properties that are common to most algorithms include:

—Longer paths disperse trust: if there is a trust path $A \rightarrow \ldots \rightarrow B \rightarrow C$, then the amount of trust inferred from A to C is not greater than the trust inferred from A to B.

—Adding paths increases trust: if there are two paths from A to B, then the trust that A infers for B is at least as high as if only one path was present.

Although intuitively sound, these properties alone are subject to shilling/Sybil attacks: in scenarios where new virtual identities can be cheaply created, a malicious node S_0 could create an unlimited number of siblings $S_1, S_2, \ldots$, add a web of strong (fake) ties between S_0 and its Sybil nodes S_i to the social network, and exploit this setup to gain a disproportionately large trust. To defend against this type of attack, trust propagation algorithms should limit the amount of trust gained by any Sybil node S_i by a function of the trust that S_0 has 'legitimately' gained.

A popular approach that guarantees the above properties is the simulation of a random walk on the web of trust, as done by PageRank [Page et al. 1998], the algorithm used by Google for ranking search results. The algorithm considers a random walk over the graph of WWW pages and their links, starting from a random node and stopping with a probability $1 - \alpha$ at each step. Nodes are then ranked according to the probability that this random walk stops at them[1]. Pages that receive many incoming links, and pages that are being linked by another heavily-linked page, are then ranked higher. Intuitively speaking, the same approach could be used to propagate trust over a social network: the higher the number of paths (equivalent to links) leading to a node (equivalent to a WWW page), the more reputable the node is assumed to be (the higher it ranks).

[1]The most common PageRank definition corresponds to the *equilibrium distribution* of a random walk, with a $1 - \alpha$ probability of jumping to a random node. The two definitions are equivalent.

The standard version of PageRank misses on subjectivity, as it ranks pages regardless of the evaluating node. As a consequence, any node in the system would propagate trust to a node X in the same way. A subjective version of the algorithm can be obtained by applying two simple changes, as already suggested in [Page et al. 1998]: first, we force the starting point of the random walk to be the evaluating node itself (thus also avoiding walks that originate at malicious nodes); second, rather than having the same probability of jumping to another node (as done in the original version of PageRank), we chose such probability to be proportional to the weight (i.e., the strength) of the edge itself. A walk starting at A will thus result in trust propagation from A's subjective viewpoint only. This modified version of the original algorithm is sometimes referred to as *Personalised PageRank* (PPR).

The original version of PageRank is subject to Sybil attacks [Friedman and Cheng 2006]. However, with Personalised PageRank, an attacker S_0 can only divert, towards the Sybil region, those paths that pass through S_0 itself. If the probability that a random walk reaches S_0 is p, then the cumulative value of all one-step paths from S_0 is αp; for two steps, it is $\alpha^2 p$, and so on. Thus, the maximal total rank for the Sybil region amounts to $\sum_{i=0}^{\infty} \alpha^i p = \frac{p}{1-\alpha}$. The α parameter thus influences the resilience to Sybil attacks: the lower the value of α, the better the robustness; if users are very careful in who they add to their social network, then trust over intent could be propagated further away (high α). However, if users carelessly add friends to their social network, then ties become less indicative of actual intent, and lower values of α (i.e., fewer propagation hops in the social network) should be preferred. Low values of α also increase subjectivity, as they reward short paths over long ones, while when α approaches 1 the outcome of the algorithm becomes more and more similar, regardless of the initiator node. PageRank is proven to have a linear convergence rate which is at least bigger than $\frac{1}{\alpha}$ [Haveliwala and Kamvar 2003]: this means that lower values of α imply faster convergence. Note, however, that low values of α may cause honest nodes who are 'socially far-away' not to be considered, thus discarding potentially useful information. This may affect the accuracy of our algorithm, with respect to traditional collaborative filtering techniques where the full dataset is considered instead.

A pseudo-code description of Personalised PageRank can be found in Algorithm 1. The computational complexity of the algorithm is proportional to the number of edges in the web of trust times the number of iterations needed to make the algorithm converge. Such complexity may hinder its practical applicability to very large networks (like the linkage structure of the web), as illustrated by means of analytical performance comparison [Haveliwala et al. 2003]. Research has thus been very active in trying to reduce the computational cost of personalised searches. Early approaches attempted to scale PPR by limiting personalisation towards a small set of representative 'topics' [Haveliwala 2003], or towards a subset of 'popular pages' [Jeh and Widom 2003]. Later approaches attempted to achieve (near) full personalisation, although at the price of precision: for example, Fogaras et al. [2005] use a Monte Carlo randomisation algorithm to approximate results over the entire web-graph; Gleich and Polito [2006] attempts to reduce the web-graph first to the small part that indeed has a non-negligible Personalized PageRank score; Gupta et al. [2008] enforce early termination of the recursive algorithm by guaran-

Algorithm 1 Personalised PageRank.

Parameters: a social network $G = (V, E)$; an evaluating node $A \in V$; weights such that w_{ij} is the weight of edge (i, j); a $0 < \alpha < 1$ parameter.
Returns: a vector r where r_i is the score of node i.

```
n ⇐ size of V; r ⇐ 0^n; r_A ⇐ 1
while algorithm has not converged do
  r̂ ⇐ 0^n; r̂_A ⇐ 1 − α
  for all i ∈ V do
    d ⇐ Σ_{j∈V} w_ij
    if d > 0 then
      for all j such that (i, j) ∈ E do
        r̂_j ⇐ r̂_j + α w_ij r_i / d
      end for
    else
      r̂_A ⇐ r̂_A + α r_i {If i is a sink we restart from A}
    end if
  end for
  r ⇐ r̂
end while
return r
```

teeing a top-k answer only (i.e., only the k nodes who can best answer the query are considered).

In this paper, we have not been concerned with scaling dependable filtering to very large social networks; investigating the problem of scalability, with potential exploitation of some the optmisations discussed above is on our agenda (Section 8). Note, however, that in the domains we target, we expect the WoT to be orders of magnitude smaller than the linkage structure of the web, thus the problem of scalability is less pressing.

An alternative way to propagate intent over a web of trust is offered by the calculation of the maximal flow function [Feldman et al. 2004], from the evaluator node A to any other node S_0. The use of the maximal flow guarantees that the inferred trust for node S_0 does not exceed the weight of any edge in the path from A to X. Such metric has been shown to be "value-sybilproof" [Cheng and Friedman 2005], that is, it is not possible for a single node S_i to obtain a trust value which is higher than the one that S_0 had before initiating the attack. Unfortunately, this result is relevant only when exactly one source node and one destination node are considered at any given time. However, in our scenario, the *overall* trust gained by the *set* of Sybil nodes must be limited too, as dependable filtering considers the judgements of many nodes simultaneously. Figure 4 shows a setup where an unlimited number of Sybil nodes S_i are linked 'in parallel' to the originator of the attack S_0: by using maximal flow on this network, each Sybil node would obtain the same (limited) amount of trust, amounting overall to an *unlimited* total flow (i.e., overall trust) for the attacking region.

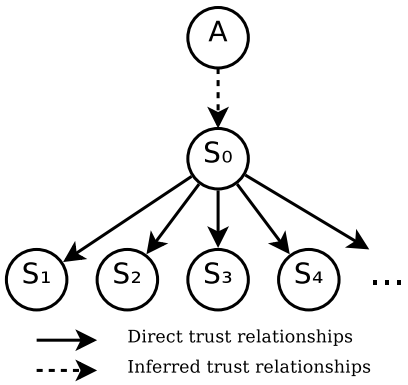


Fig. 4: "In parallel" Sybil attack to exploit maximal flow computation.

As achieving robustness is the primary goal behind dependable filtering, all the implementations of this approach that will be evaluated in the next section will use Personalised PageRank (Algorithm 1) to realise intent propagation.

4.2 How to Evaluate Concordance

The second part of our approach requires the computation of users' concordance, based on the information available in the bipartite graph of users and judgements.

Most recommender systems discussed in the literature operate on judgements that have been expressed as *numeric* ratings, explicitly assigned by users to items; in these scenarios, the aim of the recommender system is to predict ratings for yet-to-be-consumed items, and to recommend those items with the highest predicted scores. However, there exist many real-life scenarios where numeric ratings are not available, and users' preferences over items are implicitly expressed as past interactions with the system (e.g., watching a TV program, buying a book, listening to a song). These scenarios are extremely abundant, as they do not require the user to take any explicit feedback action. Recently, scenarios that were originally asking users to manually enter numeric ratings have switched to non-numeric judgements too: YouTube, for example, observed that their users were only rating videos that they liked, as shown by the overwhelming majority of 5-star ratings [Rajaraman 2009], and thus decided to drop the star-rating system in favor of a "like-dislike" one [Kofman and Rajaraman 2010]. Recommender systems developed for numeric-rating scenarios cannot be straightforwardly applied in this setting, and new techniques, known in the literature as log-based [Wang et al. 2006] or implicit-feedback [Hu et al. 2008], have started to be proposed. In realising dependable filtering, we align ourselves with this direction of research, thus dropping the assumption of having numeric ratings, and considering algorithms that can compute recommendations based on non-numeric judgements too.

We will discuss two different families of algorithms that can be used for the purpose of ranking recommendations: ranking methods and similarity-based methods, each of which is described in more details below.

4.2.1 *Ranking Methods.* Ranking algorithms realising the co-citation trust propagation pattern have been widely studied and applied to the problem of ranking Web pages. One of the most famous algorithms realising this pattern is HITS [Kleinberg 1999]. HITS conceptually divides pages in two subsets: authorities (i.e., pages whose content satisfy the query), and hubs (i.e., pages that link to relevant documents, that is, to authorities). Using an iterative process, HITS traverses the linkage structure of Web documents, and computes both a hub weight and an authority weight for each visited page at every step, so that:

(1) Forward Step (from hubs to authorities): the weight given to an authority is proportional to the sum of the weights of those hubs linking to it;
(2) Backward Step (from authorities to hubs): the weight given to a hub is proportional to the sum of the weights of those authorities being linked by it.

The process continues (renormalising scores at every iteration) until it converges, and the top ranking pages, according to their authority scores, are then returned.

The principle behind HITS is that good hubs link good authorities, and good authorities are linked by good hubs, in a mutually reinforcing way. The same principle holds in our scenario, where we can expect concordant users to give valuable judgements, and valuable judgements to be given by concordant users. If we map users to hubs and judgements to authorities, we can run an HITS-like iterative algorithm to compute concordance scores on users and to rank judgements, which is our ultimate goal. If weights expressing concordance are present in the network of judgements, they can be used as a multiplicative factor (i.e., a link with weight 2 acts as two separate links, each with weight 1). However, some adaptations to HITS are necessary in order to fully realise the dependable filtering approach.

(1) Solving the TKC Problem. It has been demonstrated that the HITS algorithm suffers from the "Tightly Knit Community" (TKC) syndrome [Lempel and Moran 2001]: if a community of users all gave the same (or very similar) judgements (thus resulting in a highly connected bipartite graph), the concordance weight of the community would disproportionately increase, with the judgements they express being excessively high-ranked, even if they are not authoritative. A set of malicious users could thus artificially create a TKC in order to artificially boost their ranking. To solve this problem, the same technique adopted in SALSA [Lempel and Moran 2001] can be exploited: we divide the weight that each hub transfers at each forward step by its outdegree (the sum of weights on outgoing edges), and we do the same for authorities and their indegree at each backward step. After a forward step, the total weight transferred from a single hub to its linked authorities is thus equal to the weight on that hub; vice versa, after a backward step, the total weight that is redistributed from a single authority to the set of hubs linking to it equals the weight gained by the authority. Thus, the sum of weights remains constant at every step, removing the need for normalization. The result is a PageRank-like random walk on the bipartite graph of users and judgements. A very desirable side-effect of this alteration is that users who agree on "niche" judgements are rewarded more than those expressing only mainstream (redundant) ones: if a user, in her top 10 listenings, has both The Beatles and an unheard-of rock band, it is likely that the unknown rock band is more indicative of her musical preferences. Moreover, an attacker would not be rewarded for copying very popular judgements in the hope of

raising their concordance ranking on many nodes. This is analogous, in principle, to the TF-IDF approach used in text retrieval, where the significance of a term in a document is determined by multiplying the number of occurrences in the document with the *inverse* of the overall frequency of that term.

(2) Subjectivity of Ranking. HITS-like algorithms provide non-subjective results, as they are independent of the user A starting the search. To cater for the subjectivity required by our scenario, we initialize the algorithm so that the only hub (user) with a non-zero weight is the reference node A itself (instead of assigning an equal weight to any hub in the network). In so doing, the first forward step of the algorithm only considers the judgements given by the reference node, thus tailoring the ranking results to his/her tastes. To limit the propagation of trust to judgements that are too dissimilar from the tastes of A, after each backward step, the weights associated to each user are multiplied by a parameter $\beta \in (0, 1)$, and the trust given to A is increased by $1 - \beta$. These two changes are similar, in spirit, to the modifications already suggested for PageRank, where we forced the random walk to start from the very same node; the β parameter plays the same role that α plays in PageRank, ensuring the convergence of the algorithm, with lower values of β implying faster convergence and higher subjectivity. As observed by Huang and Zeng [2005], this is particularly desirable in datasets which exhibit a high clustering coefficient in the users/judgements bipartite graph.

It is worthwile to note that the TF-IDF adjustment is actually used in recommender systems and it is known to enhance recommendation quality [Deshpande and Karypis 2004]; the usage of this method can be motivated with a probabilistic justification [Wang et al. 2006]. These recommender systems can be seen as close relatives of our own when trust on concordance is only propagated one step (i.e., $\beta \to 0$).

4.2.2 *Similarity-based Methods.* An alternative approach to ranking algorithms is given by similarity-based approaches, that have been widely used in collaborative filtering (CF) realisations of recommender systems. The underpinning assumption is that users who have been like-minded in the past will most likely be so in the future. These techniques thus compute concordance as a similarity score between each pair of users, based on the items they have co-rated; these scores are then used to weight recommendations. Many of them require judgements to be in the form of numeric ratings, such as in the case of Pearson correlation and its variants; the interested reader may find an extensive analysis and comparison between these measures in the work of Lathia et al. [2008b]. As a baseline comparison, we will instead consider cosine similarity, which can be directly applied also to sets of non-numeric judgements by considering vectors of binary values where each value is either a zero (not given) or a one (given) for each possible judgement. Despite its simplicity, the performance results of cosine similarity have been proven consistently good [Lathia et al. 2008a].

Note that various other techniques exist in the literature that work on the bipartite graph of users/judgements, mainly to improve the *accuracy* of collaborative filtering when faced with severe sparsity problems. For example, Aggarwal et al. [1999] define a graph-theoretic approach capable of providing accurate recommendations in sparse settings, and despite potential misalignments of users' rating scales,

or indeed different tastes (i.e., opposite judgements). Huang et al. [2005] suggest using link predictions algorithms, adapted from the social network domain to the bipartite graph of users/items; Huang et al. [2004] exploit spreading activation algorithms, which they extensively evaluate on e-commerce data; Huang et al. [2007] use a graph partitioning technique which ensures high accuracy in scenarios where a high clustering coefficient in the user/item bipartite graph exist. To enable the comparison, in terms of recommendation accuracy, between such techniques and dependable filtering, we have implemented a "branch and bound" algorithm belonging to the spreading activation model family [Huang et al. 2004], and used it as a benchmark in Section 7.1.

4.3 Dependable Filtering Algorithms

In order to realise the Dependable Filtering philosophy, algorithms for computing intent and concordance must be combined. As robustness is a primary concern, we restrict our attention to PPR for computing intent. In computing concordance, both approaches based on random walks and those based on similarity measures are worth evaluating, giving rise to the following four combinations:

(1) PPR + RNDWALK;
(2) PPR + Cosine;
(3) PPR + [Cosine + Iter], and
(4) PPR + [Cosine + Iter + γ].

[Full Realisation 1] Dependable Filtering = PPR + RNDWALK. By combining PPR (trust over intent) and the random walk of the modified HITS (trust over concordance) we obtain the first full realisation of dependable filtering. It is worth noting that HITS-like algorithms would already return a ranking of judgements, but they do so in a way that is susceptible to attacks, as discussed in Section 3. To add robustness to HITS-like algorithms and thus realise dependable filtering, we incorporate users' intent assessment as follows. To begin with, Personalised PageRank is run on the social network, thus obtaining a vector with nodes' reputation, as seen by the reference node A. We then run the subjective HITS-like algorithm (RNDWALK), so that, at every backward step, trust is redistributed from judgements to users in a way that is *proportional to users' intent*, as measured by PPR. In other words, *reputation becomes a multiplicative factor for backward trust propagation*. As discussed in Section 4.1, a Sybil coalition can obtain only a limited amount of trust from the social network, so the amount of trust that can be transferred to malicious nodes is limited too. The resulting pseudocode is shown in Algorithm 2: the result of running PPR+RNDWALK is a vector $\hat{t}$ containing a *trust* numeric value for each judgement in J, computed considering both the intent and the concordance of the users in U, as seen by the reference node A. The normalization parameters ($\sum_{k \in J} w_{uk}$, $\sum_{v \in U} w_{vj} r_v$) can be calculated outside the loops, so the computational cost of the full algorithm is proportional to the number of edges in the network of judgements times the number of iterations.

Note that RNDWALK can be seen as an instance of Personalized PageRank, where the weight w_{ij} of the edge between users i and j is defined as the product $p_{ij} \cdot r_j$, where p_{ij} is the probability that a two-step random walk on the user-

Algorithm 2 Dependable Filtering: PPR + RNDWALK.

Parameters: a network of judgements $G = (V, E)$, where V is the union of the set of users U and the set of judgements J; an evaluating node $A \in U$; weights such that w_{uj} is the weight of edge (u, j); an intent ranking vector r computed using Personalised PageRank over the web of trust, so that r_u is the intent ranking of user u; a $0 < \beta < 1$ parameter.
Returns: a trust vector $\hat{t}$ such that $\hat{t}_j$ is the trust ranking of judgement j.
$n \Leftarrow$ size of U; $m \Leftarrow$ size of J; $t \Leftarrow 0^n$; $t_A \Leftarrow 1$
while algorithm has not converged **do**
  {Forward Step: from users to judgements}
  $\hat{t} \Leftarrow 0^m$
  for all $(u, j) \in E$ **do**
    $\hat{t}_j \Leftarrow \hat{t}_j + \frac{w_{uj}}{\sum_{k \in J} w_{uk}} t_u$
  end for
  {Backward Step: from judgements to users}
  $t \Leftarrow 0^n$; $t_A \Leftarrow 1 - \beta$
  for all $(u, j) \in E$ **do**
    $t_u \Leftarrow t_u + \beta \frac{w_{uj} r_u}{\sum_{v \in U} w_{vj} r_v} \hat{t}_j$
  end for
end while
return $\hat{t}$

judgements network that starts from node i will arrive to node j, and r_j is the intent ranking of node j. In this case, the β parameter plays the role of α, and the convergence rate for RNDWALK is thus at least $\frac{1}{\beta}$ due to the convergence properties of PPR mentioned in Section 4.1.

An advantage of performing a random walk to evaluate trust over concordance is its ability to *propagate* it: if A has shared judgements with B, and B has shared further judgements with C, HITS propagates concordance from A to C, so that C's judgements are considered when creating recommendations for A (obviously discounted by the factor β). This becomes particularly important in sparse datasets, where users share very few judgements. An objection that may be raised though is that trust over concordance is highly subjective, and thus not transitive (as it would be for intent instead); however, in situations where lack of data prevents other techniques from being used, intent propagation appears to be a valuable bootstrapping technique, as our experimental results will demonstrate.

[Full Realisation 2] Dependable Filtering = PPR + Cosine. By combining PPR (trust over intent) and cosine similarity (trust over concordance) we obtain a second realisation of dependable filtering. The overall approach would perform the following steps: as before, we incorporate users' intent assessment by first running Personalised PageRank on the social network, thus obtaining a vector with nodes' reputation, as seen by the reference node A. We then compute concordance between A and any other node X as the cosine similarity on the judgements they have expressed; the relevance (weight) of a judgement expressed by X would then

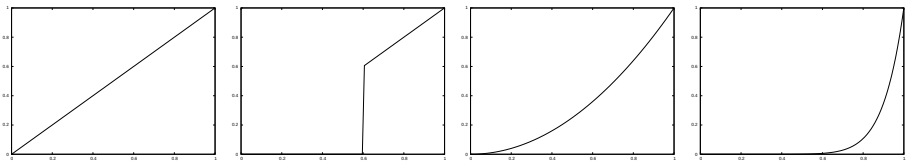
Algorithm 3 Dependable Filtering: PPR + [Cosine + Iter].

Parameters: the set of users U; the set of judgements J; an evaluating node A; an intent ranking vector r computed using Personalised PageRank over the web of trust, so that r_u is the intent ranking of user u; the number n of iterations; a $0 < \beta < 1$ parameter.
Returns: a trust vector $\hat{t}$ such that $\hat{t}[j]$ is the trust ranking of judgement j.
Create a matrix t such that $t[u][j] = 1/k$ if user u has expressed judgement j, and 0 otherwise, with k being the number of judgements expressed by u
$\hat{t} \Leftarrow t[A]$
while algorithm has not converged **do**
  {Compute correlations based on similarity and intent}
  for all $u \in U$ **do**
    $C[u] = \cos(\hat{t}, t[u]) \cdot r_u$
  end for
  {Update the rankings}
  for all $j \in J$ **do**
    $\hat{t}[j] = \sum_{u \in U} t[u, j] \cdot C[u]$
  end for
  $\hat{t} \Leftarrow \frac{\hat{t}}{|\hat{t}|_1}$ {Normalize}
  $\hat{t} \Leftarrow (1 - \beta) \cdot t + \beta \cdot \hat{t}$ {Iterate}
end while
return $\hat{t}$

simply be X's intent multiplied by its concordance. To defend against malicious nodes that, in order to increase their influence on honest nodes, express a disproportionally large number of judgements, the weight associated to a judgement is divided by the number of judgements a node has expressed.

[Full Realisation 3] Dependable Filtering = PPR + [Cosine + Iteration]. Similarity-based approaches are simpler than ranking algorithms, and have proved to work well on dense datasets. This may not be a problem when computing recommendations for users with mainstream tastes; however, as the long tail phenomenon grows, sparsity is likely to characterise most situations, with consequent impact on coverage. In these circumstances, propagation of concordance is called upon. We thus propose a further realisation of dependable filtering that *iteratively propagates* concordance values computed using cosine similarity. More precisely: the user A starts a trust vector t such that t_j is the weight she gave to the judgement j; using the approach outlined before (realisation 2), we obtain a new vector $\hat{t}$ giving new weights to all judgements. We then iterate the algorithm by updating the values in t, such that the new value of the t vector is equal to $(1 - \beta) \cdot t + \beta \cdot \hat{t}$. The β parameter plays essentially the same role it played in the PPR+RNDWALK algorithm, with higher values of β giving more relevance to concordance propagation. The resulting algorithm can be found in Algorithm 3.

[Full Realisation 4] Dependable Filtering = PPR + [Cosine + Iteration + γ]. Various improvements can be made to dependable filtering realisations based on similarity measures. For example, in the domain of traditional recommender sys-



(a) Linear. (b) Threshold. (c) $\gamma = 2$. (d) $\gamma = 10$.

Fig. 5: Weighing the correlation. The four plots show weight given to judgements (Y axis) versus correlation (X axis). Linear weighting is the default, while threshold techniques zero out the contributions given by users with correlation below a given threshold, which is either statically or dynamically (kNN, kNR) computed. Our technique enhances the relevance given to highly similar nodes, modulating this effect with the γ parameter.

tems, k-Nearest Neighbours (kNN) and k-Nearest Recommenders (kNR) algorithms have been successfully applied. Their goal is to identify the best recommenders in the set and rely on them only. On sparse datasets, cutting the number of recommenders in this way may have detrimental effects on coverage. In order to gain in accuracy without compromising coverage, we propose the following algorithm tuning: users' correlation (computed using cosine similarity) is magnified by raising it to the power of a positive constant $\gamma > 1$, in order to mark differences between good and bad recommenders neatly. In so doing, judgements made by good recommenders become sensibly higher ranked than others (see Figure 5), although no judgement made by a non-zero similarity node is ever discarded (as it would happen for kNN or kNR). This dependable filtering algorithm would thus be the same as the one shown in Algorithm 3, with the difference that $C[u] = \cos(\hat{t}, t[u]) \cdot r_u$ now becomes $C[u] = \left(\cos\left(\hat{t}, t[u]\right)\right)^{\gamma} \cdot r_u$.

In a preliminary version of this paper [Dell'Amico and Capra 2008], some experimental results based on PPR + RNDWALK (full realisation 1) have been presented; in this paper, we thus focus our attention more on the three new variants.

5. ATTACK MODEL

In order to evaluate our dependable filtering algorithms, we have conducted a variety of experiments on two very large real datasets. While ideal to measure accuracy, real datasets are unsuitable to test the robustness of the algorithms while varying threat intensity. To demonstrate the robustness of dependable filtering, we thus have to manually inject attacks on top of real datasets, and run experiments under different configuration settings. In this section, we analyse threat strategies, leaving their enactment and corresponding experimental validation to Section 7.

In the scenario we are considering, the most plausible goal of an attacker would be to alter the rating of a certain judgment X. It may do so either to trick a single user A, or more extensively to deviate the judgments of all users, in favour of (or against) X. Let us analyse how an attacker could achieve such goal. In the first case, since the attacker wants to be rated by A as a very concordant user, it could first copy the judgments that A expressed, and then add a new judgment X. In the second case, there is no single set of judgments the attacker can copy, as each user would have expressed different ones. We will thus model this attack by copying the

judgments of a randomly chosen node and adding the judgment for X; we will then measure the ranking of X *on a random node* in the network. We point out that there is no obvious way of obtaining a higher attack impact on the whole network: if the attacker copies popular judgements, it will be deemed similar to many users, but it will have to compete with many of them for a high ranking; if 'niche' judgements are copied, a higher appeal will be obtained towards a small number of users. Also, if an attacker copies a set of highly-correlated judgements, then this will result in high appeal for the users that actually expressed those correlated judgements, but not much influence on other users of the system.

The attack strategies described above model the behavior of one attacker only. However, to increase the impact of the attack itself (i.e., to increase the ranking of judgment X), we must also consider the case of an attacker who has the ability to create an unlimited number of Sybil identities, all endorsing X. We assume that each Sybil can create any number of outgoing edges in the web of trust, from the Sybil node to any other user. They can also create any number of incoming edges, originating within the Sybil coalition. However, what they cannot do is create incoming edges from honest nodes at will, since obtaining trust from well-intentioned peers is costly. It is thus reasonable to expect a low cut between the "honest" and the "Sybil" region; the links between these two parts of the graph are called *attack edges* [Yu et al. 2006]. In our experiments, we will thus create Sybil regions that are highly interconnected internally; we will then set the amount of incoming links from honest nodes as a parameter, and analyse the robustness of dependable filtering algorithms (i.e., how highly ranked can X become) against it.

6. EXPERIMENT SETUP

We have evaluated dependable filtering along two dimensions: accuracy and robustness against Sybil attacks. The results are reported in section 7.1 and 7.2 respectively. Both experiments were conducted using data from two real datasets: the Citeseer online scientific digital library, and the Last.fm music and social networking website. The key characteristics of these datasets, the simulation parameters and evaluation metrics are briefly summarised below, before presenting our actual results.

6.1 The Datasets

Citeseer (`http://citeseer.ist.psu.edu/oai.html`). An online scientific literature digital library, containing over 750,000 documents. From this repository, we have extracted a social network based on the co-authorship relation: if A and B have co-authored n papers together, then an edge between the two will be added to the social network, with weight n. The bipartite judgement network is built from the citations instead: if a paper X cites paper Y, then an (unweighted) edge from X to Y is added to the judgment network; the rationale is that, by citing Y from X, the authors have expressed the judgement "Y is relevant with respect to the topic we discuss in X". In so doing, a paper can appear in the bipartite judgement network as both a source node (i.e., paper making a citation) and as a destination node (paper being cited); in this case, two different nodes will represent the same paper in the two different roles.

We need an adjustment to make our recommendation algorithms work on this

dataset: the PPR algorithm computes an intent trust value for authors of papers from the social network of co-authorships; however, authors do not appear in the judgement network, as the origin of judgements are papers. We thus need to propagate trust from authors to papers; we do so in the following way: if an author A has a trust value t_A and she has authored n papers, the intent trust value propagated to each paper is t_A/n, in order to ensure that very prolific authors do not have an influx on recommendations which is disproportionate to their trust value. Furthermore, if a paper has more than one author, its overall trust is the sum of the intent trust values coming from each of its authors. For example, if Alice has 4 papers in the dataset and her intent trust value t_A (computed with PPR on the co-authorship network) is 0.08, and Bob has an intent trust value $t_B = 0.15$ and 3 papers in the dataset, the intent trust value for a paper co-authored by the two of them will be $t_A/4 + t_B/3 = 0.07$.

To obtain a more manageable subset of the whole network, we isolated a highly-clustered subset of 10,000 authors[2], and took in consideration only the papers that had them as authors. The result is a set of 182,675 different papers and 447,715 citations; 48,998 papers in the dataset received at least one citation. The average number of connections per node is 16.75, and the average weight on each edge is 3.87.

Last.fm (`http://last.fm/`). A "social music" website that creates profiles of the musical taste of its users, by tracking which songs they listen more often to in their digital music players. As in other social networking websites, users can explicitly create an (unweighted) social network by adding other users to their friend-list. We gathered our social network with a breadth-first crawl of 10,000 users using the Audioscrobbler Web Services (`http://www.audioscrobbler.net/data/webservices/`). We then considered the 50 most listened artists of each user, and ended up with a total of 51,654 different artists. The judgment network was finally created by linking users to their most listened artists (thus representing the judgment "user A likes to listen to songs by X"), and by weighting each judgment edge with the number of times the user listened to songs by that artist. The average degree of the web of trust is 7.25.

6.2 Metrics

We claim that our dependable filtering algorithms are able to give high quality recommendations to users, without falling prey to Sybil attacks. To validate this statement, we have evaluated both *accuracy* and *robustness* as detailed below.

6.2.1 *Accuracy: Ranking Hidden Judgements.* When evaluating Collaborative Filtering (CF), reference datasets are usually composed of judgments containing a numeric value, for example, a rating in a zero to five stars scale. CF algorithms assemble those ratings and try to predict how a user would rate a given object such as a book or a film. In that case, evaluating accuracy is straightforward: numeric judgments are removed from the dataset and then the CF algorithm is used to

[2]We obtained our Web-of-Trust (WoT) subset by starting with a random author, and then simulating a crawl that iteratively explored the node that had most co-authorships with already-crawled authors.

predict the missing ones. The most used measure to assess accuracy is the Mean Absolute Error (MAE) of the prediction.

In the Last.fm and Citeseer datasets, though, only one kind of judgment can be present for any given object, be that a paper (e.g., paper X has been referenced, and thus is deemed relevant) or an artist (e.g., I listen to music of artist X, thus I like X); in both cases, there is no numeric rating associated to the judgement, and thus no numeric value to be predicted.

To assess the accuracy of dependable filtering algorithms in giving recommendations, we performed the following experiment instead on both datasets: we "hid" one random edge $A \rightarrow X$ from the judgment network, ran our algorithms on the modified network, and used their output (i.e., a vector of weights) to rank all judgments from A's viewpoint; this is equivalent to producing recommendations, tailored to A, based on the computed ranking of judgments. Since X is a judgment that A expressed (before we hid it), A obviously approves of it, so a good recommendation engine should return X at a very high ranking. Thus, the highest the position of X in the ranked list of judgments, the better the accuracy of the ranking algorithm. In the Citeseer dataset, the experiment is equivalent to guessing a missing citation from a paper; in Last.fm, it means finding the missing artist in the top-50 chart of a user.

For recommender systems based on non-numeric judgements, it is customary to create a training set and a test set of judgements, and then verify whether, given a top-k list of recommendations, the items in the test set appear within the recommendations (see e.g. Deshpande and Karypis [2004] and Huang et al. [2004]). There is a natural mapping to our metric: for each instance of the evaluation, we consider a single judgement to be part of the test set, with everything else being in the training set; we then evaluate the probability that the missing judgement is within the top k recommendations as k varies.

6.2.2 *Robustness: Ranking Malicious Judgements.* As discussed in Section 5, we are interested in evaluating how much an attacker, by creating a very high number of fake nodes, can raise the ranking of a given judgment X. We have thus designed our experiments as follows: we create a completely connected Sybil sub-network, and attach it to the honest part of the web of trust with a parametric number k of attack edges; each attack edge is given a weight of 1, and the honest node to which it connects is chosen at random. In the *untargeted* attack, where the goal for the attacker is to generically raise the ranking of the malicious judgement on any node in the network, each Sybils thus copies all judgements given by a random peer and adds the malicious judgement (giving it the maximal weight between the copied judgements). In the second case, the *targeted* attack, where the objective is specifically to raise the ranking of the malicious judgement for a *victim* node, the Sybil nodes always copy the judgements of the victim, instead of choosing peers at random.

In our experiments, we study how the ranking of the malicious judgement varies (on a random node in the network in the case of untargeted attacks, and on the victim in the case of targeted attack), *for different values of* k.

Note that the strength of an attack on traditional CF techniques has usually been measured in terms of the proportion of malicious nodes in the network [Mobasher

et al. 2006]; however, the number of Sybil nodes is not relevant for our dependable filtering algorithms based on PPR, where the impact of the attacker is limited by the total ranking of the Sybil region. We thus not focus on the number of Sybils and instead vary parameter k, which does influence the ranking of the Sybil region instead.

6.3 Simulation

All results presented in the next section are given after at least 1,000 individual instances of the experiment for each choice of dataset, algorithm, and set of parameters. If not otherwise mentioned, the Sybil region is composed of 100 nodes in the malicious judgement experiment.

In both cases of hidden and malicious judgements, we are interested in studying the ranking of those judgements using the various algorithms proposed; we did this by studying their cumulative distribution function (CDF), that is, the probability that the hidden/malicious judgement is found at position x or less. In addition to plotting the CDFs, we summarize the results with table containing the percentiles: for example, a value of x for the 10th percentile means that 10% of the times the judgement is ranked at position x or less. An equivalent way to see the plots is as "hit-rate" (y axis) versus length (x axis) of the recommendation list, as defined by Deshpande and Karypis [2004].

7. RESULTS

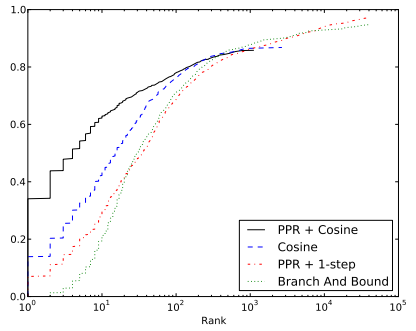
In this section, we present the results obtained while conducting an extensive set of simulations. First, we report on the accuracy of the various dependable filtering algorithms proposed in the absence of attacks; second, we demonstrate their resilience in the face of attacks of different magnitude.

7.1 Accuracy

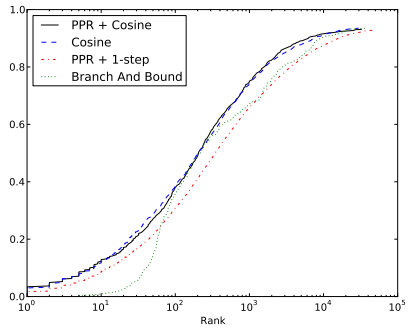
7.1.1 *Introducing Dependable Filtering.* To evaluate the effectiveness of the dependable filtering concept, we started by comparing its simplest realisation ("Realisation 1 = PPR + Cosine") with its constituents, that is, transitive trust propagation over the WoT only, and co-citation trust propagation over the judgements graph only. The former has been implemented simply as PPR on the user network, with the so computed intent ranking equally divided among all the judgements that that node expressed ("PPR + 1-step"). The latter has been implemented as standard cosine similarity ("Cosine").

To provide a baseline comparison with a known algorithm that explores the network of judgements, we also implemented a "branch and bound" algorithm based on spreading activation models [Chen and Ng 1999] on the judgement network[3]. This algorithm has been applied to collaborative filtering in order to cope with sparse datasets (as in our case) and it applies to non-numeric judgements [Huang et al. 2004]. This algorithm performs a visit on the network of judgements, using a priority policy which ensures that more "promising" (i.e., better connected to the

[3]To ensure a preference for shorter paths, the algorithm requires edge weights to be lower than 1. We accomplish this by renormalizing weights so that the sum of outgoing edges from each "user" node is 1.



Algorithm	Percentiles		
	10	50	90
PPR + Cosine	1	4	–
Cosine	1	15	–
PPR + 1-step	2	35	3,153
Branch And Bound	6	30	2,878



Algorithm	Percentiles		
	10	50	90
PPR + Cosine	7	206	6,239
Cosine	8	204	7,735
PPR + 1-step	12	344	16,025
Branch And Bound	41	227	9,963

Fig. 6: Hidden judgement ranking: introducing dependable filtering on Citeseer (left) and Last.fm (right).

origin) nodes are visited first; the recommendation accuracy obtained is known to be in line with other algorithms based on spreading activation models. Figure 6 illustrates the results obtained.

As shown, on the Citeseer dataset (left), the median (i.e., 50th percentile) ranking for the hidden judgement is as high as 35 for "PPR + 1-step" and 15 using cosine similarity, while it drops to 4 with the dependable filtering ("PPR + Cosine") approach. Of particular relevance is the observation that, even now that malicious attacks are *not* considered, dependable filtering outperforms standard CF, despite the fact that it throws away (potentially useful) information coming from (honest) socially far-away nodes. This means that dependable filtering effectively exploits knowledge gathered from the social network to counter-balance this loss of data, and the gain is higher than the cost for datasets that, like Citeseer, exhibit an intrinsic "homophily" property: neighbours in the social network are more likely to share tastes. The simple PPR-based algorithm, ignoring information about judgements, as expected performes worse. We also notice that neither Cosine nor PPR + Cosine are able to give a nonzero ranking for the "most difficult" 15% of judgements, while basic PPR can do this by virtue of the fact that trust is propagated transitively to all reachable nodes in the web of trust. As we shall demonstrate later, more advanced realisations of dependable filtering that do propagate trust over concordance will remove this limitation. We also notice that, albeit providing high coverage (i.e., being able to rank many judgements), the branch and bound algorithm does not perform well in terms of recommendation quality. Since recommendations are provided during the process of graph exploration, the first ones are only based on a very limited amount of information, and this negatively affects their quality. Our dependable filtering approach instead provides better recommendations at the expense of a higher computational cost.

Percentile	5	10	25	50	75	90
Citeseer	1 – 1	1 – 1	1 – 1	4 – 5	35 – 65	–
Last.fm	2 – 4	7 – 11	36 – 61	182 – 305	1,005 – 1,650	6,239 – 15,180

Table I: Minimum and maximum values for hidden judgement ranking with varying $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$. Algorithm: PPR + Cosine.

Percentile	5	10	25	50	75	90
Citeseer	1 – 1	1 – 1	1 – 1	4 – 5	34 – 54	437 – 1,385
Last.fm	2 – 3	7 – 9	35 – 51	190 – 246	1,119 – 1,422	4,828 – 16,166

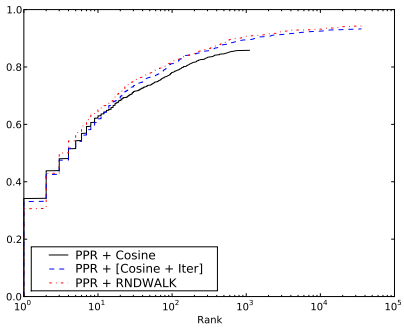
Table II: Minimum and maximum values for hidden judgement ranking with varying $\beta \in \{0.1, 0.2, \ldots, 0.9\}$. Algorithm: PPR + [Cosine + Iter].

On the Last.fm dataset (Fig. 6 right), the accuracy of PPR + Cosine is roughly equivalent to the one of Cosine alone, meaning that the loss of information due to filtering nodes is counterbalanced by the benign effect of taking into account user homophily in the social network. Again, recommendations produced by the Branch And Bound algorithm are not on par with the other algorithms, especially with respect to the highest-ranked recommendations.

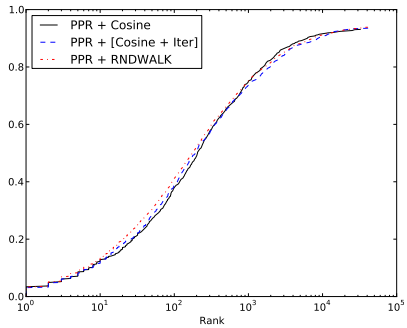
The above experiments have been conducted with $\alpha = 0.5$ for Citeseer and $\alpha = 0.3$ for Last.fm. In order to test the sensitivity of our algorithm to the parameter, we have performed a number of experiments, with $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$; Table I summarizes the minimum and maximum values for the hidden judgement ranking when changing α. As shown, accuracy is not significantly impacted by this parameter; we have thus set $\alpha = 0.5$ for Citeseer and $\alpha = 0.3$ for Last.fm in the reminder of the experiments we report on, as these values provide consistently good accuracy independently of the choice of other parameters.

7.1.2 *Iterative Trust on Concordance.* The previous set of experiments shows, on two very different datasets, that the simplest realisation of dependable filtering, that is the PPR + Cosine algorithm, is able to perform with an accuracy which is comparable to, or better than, the cosine-based traditional collaborative filtering algorithm for the more mainstream/popular judgements. For more "difficult"/niche ones, however, accuracy can drop due to the fact that not enough information is available to compute a prediction, since both intent ranking and correlation are needed in order to trust judgements made by other peers. This can be viewed as akin to a "low coverage" problem in traditional CF settings. To address the issue of coverage, we have introduced in Section 4.2 algorithms that *iteratively* propagate concordance over the network of judgements. Figure 7 compares the performance of the simple PPR + Cosine algorithm, with the two iterative realisations of dependable filtering, that is, "PPR + [Cosine + Iter]" and "PPR + RNDWALK". As shown, iterative concordance propagation neatly increases the accuracy and coverage of the harder-to-predict judgements (50-100 percentiles) on Citeseer (left). In Last.fm (right), where coverage was not a problem, introducing iterative trust propagation does no harm, leading to results that are better than, or comparable to, traditional collaborative filtering across all scenarios.

Similarly to what we have done with respect to α, we show in Table II the



PPR + ...	Percentiles		
	10	50	90
Cosine	1	4	–
Cosine + Iter	1	4	1,385
RNDWALK	1	4	855



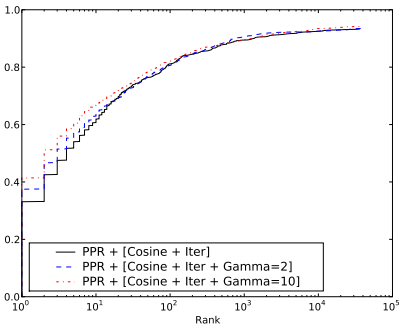
PPR + ...	Percentiles		
	10	50	90
Cosine	7	206	6,239
Cosine + Iter	8	193	8,871
RNDWALK	6	174	7,429

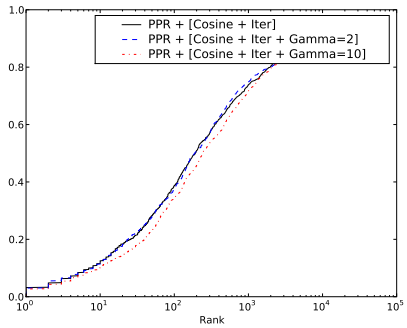
Fig. 7: Hidden judgement ranking: iterative trust on concordance in Citeseer (left) and Last.fm (right).

impact of a varying $\beta \in \{0.1, 0.2, \ldots, 0.9\}$ in our experiments. Once again, the recommendation quality is not very sensitive to variations in β. We empirically adopted a value of $\beta = 0.3$ for Citeseer and $\beta = 0.05$ for Last.fm, values which provided us good results in term of accuracy.

7.1.3 *The γ Parameter.* To enhance the significance given to highly correlated peers, we experimented with various values of γ; results are reported in Figure 8. As shown, the impact of the parameter varies wildly depending on the dataset at hand: in Citeseer (left), the median ranking of the hidden judgement drops from 4 to 2 when γ is raised to 10; on Last.fm (right), the effect is instead only marginally beneficial when $\gamma = 2$, and sensibly detrimental when $\gamma = 10$. This difference is related to the intrinsic characteristics of the datasets: a high similarity in citations between two different research papers is a very strong indicator of similarity in the topic discussed by the two papers; on the other hand, high similarity in the most-listened charts appears to be less indicative, and a $\gamma = 10$ parameter will rely too much on users with higher correlation, underestimating information coming from others.

In Figure 9, we compare our use of the γ parameter against the more widely known kNN technique [Herlocker et al. 1999], focusing on the Citeseer dataset where this kind of optimization is actually beneficial. The k parameter involves a tradeoff between number of ranked judgements ("coverage") and quality of the results: when k is low there is a higher recommendation quality but coverage is smaller, because only the items rated by the nearest k neighbors can be rated; as k grows, the benefit obtained by the for the "easier" judgements is lost. Our method, instead, consistently provides better accuracy than kNN for any choice of parameters, whilst not sacrificing on coverage, as no information needs to be discarded.

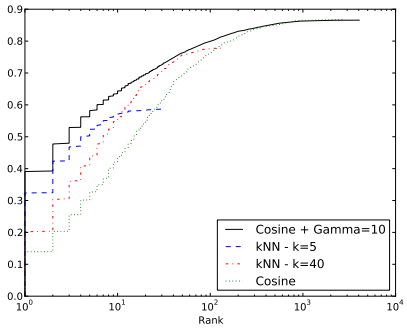




γ	Percentiles				
	10	25	50	75	90
1	1	1	4	41	1,385
2	1	1	3	41	761
5	1	1	3	32	1,100
10	1	1	2	33	1,280
20	1	1	3	45	3,724

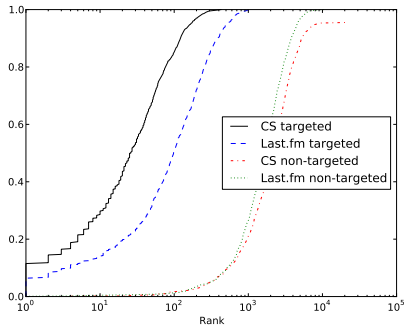
γ	Percentiles				
	10	25	50	75	90
1	8	41	193	1,118	8,871
2	8	40	190	1,036	12,344
5	9	56	263	1,037	7,008
10	10	57	242	1,296	9,497
20	10	61	295	1,253	5,467

Fig. 8: Hidden judgement ranking: adding γ in Citeseer (left) and Last.fm (right).



Percentiles	10	25	50	75	90
Cosine + $\gamma = 10$	1	1	3	41	–
kNN – $k = 3$	1	1	–	–	–
kNN – $k = 5$	1	1	4	–	–
kNN – $k = 10$	1	1	4	–	–
kNN – $k = 20$	1	2	5	–	–
kNN – $k = 40$	1	2	7	46	–
kNN – $k = 80$	1	2	8	44	–
kNN – $k = 160$	1	3	10	48	–
Cosine	1	3	15	88	–

Fig. 9: Hidden judgement ranking: kNN and Cosine + γ on Citeseer.



Percentiles	10	25	50	75	90
Citeseer targeted	1	7	26	62	122
Last.fm targeted	4	33	98	219	388
Citeseer non-targeted	530	1,161	2,141	3,483	5,338
Last.fm non-targeted	518	940	1,622	2,656	3,784

Fig. 10: Malicious judgement ranking: PPR + Cosine + Iter on Citeseer and Last.fm.

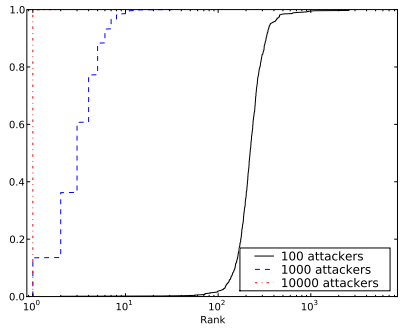
7.1.4 *Lessons Learned.* The experimental assessment of accuracy reported in this section has highlighted that different realisations of dependable filtering produce different levels of accuracy that are strongly coupled with the characteristics of the dataset used. Whilst there is no clear winner between the examined algorithms, it distinctly emerges that dependable filtering is consistently able to give recommendations with an accuracy which is comparable to, or better than, state-of-the-art collaborative filtering techniques.

Having demonstrated that accuracy is not hindered by exploiting information coming from the social network, we now move on to evaluate the added value that dependable filtering brings in terms of robustness and higher resilience to attacks.

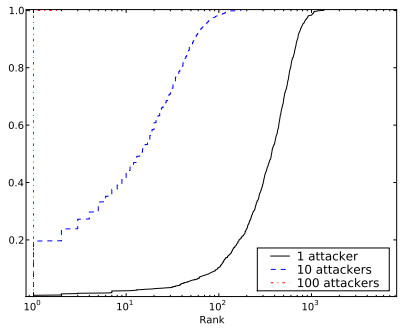
7.2 Robustness

In Figure 10, we show the results for malicious judgement ranking on our two datasets against the PPR + Cosine + Iter algorithm. The targeted attack is more successful in the Citeseer case; we attribute this phenomenon to the fact that citations in a paper are more predictable than the artists a Last.fm user could listen to and thus a set of ad-hoc judgements is more likely to influence the resulting recommendations. Besides this one effect, we have found that the results obtained in terms of robustness are qualitatively equivalent on the various datasets we used. In the interest of terseness, in the following we will therefore only report results for the Last.fm dataset.

7.2.1 *Attacks on CF Algorithms.* As a first experiment, we assessed the impact of the attack we devised on traditional CF algorithms. As Figure 11 illustrates, the strength of the attack strictly depends on the number of attackers. In particular, the untargeted attack (left) becomes strong when the number of Sybils is comparable with honest users; however, 100 nodes (roughly 1% than the number of honest users) are enough to always put the malicious judgements at position 1 for targeted attacks. In scenarios where new fake identities (Sybils) can be cheaply created, CF



Attackers	Percentiles				
	10	25	50	75	90
100	152	184	221	266	330
1,000	1	2	3	4	6
10,000	1	1	1	1	1



Attackers	Percentiles				
	10	25	50	75	90
1	98	205	371	554	709
10	1	3	14	34	55
100	1	1	1	1	1

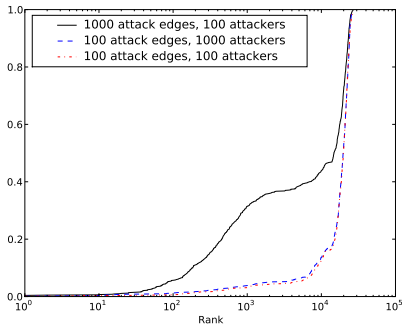
Fig. 11: Malicious judgement ranking: cosine-similarity CF, with non-targeted (left) and targeted (right) attacks.

approaches thus do not offer sufficient resilience to attacks.

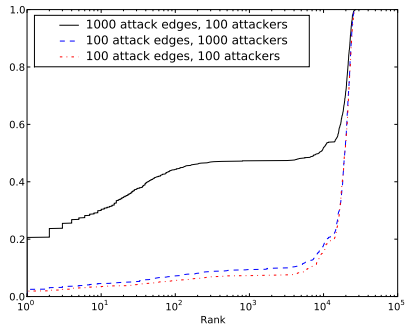
7.2.2 *Robustness of Dependable Filtering.* Contrary to collaborative filtering, dependable filtering is largely insensitive to the number of attackers: as Figure 12 illustrates, the ranking of the malicious judgement remains basically unchanged when the Sybil attackers (n) vary from 100 to 1,000. Rather, the attack strength should now be measured against the number of honest users tricked by an attacker into trusting her, which is a much harder task than creating new fake identities. In the reminder of this section, we thus analyse the susceptibility of the various protocol variants to attacks where a Sybil region of $n = 100$ fake nodes has managed to create $k = 100$ attack edges towards the honest part of the network.

Figure 13 compares the resilience of the basic "PPR + Cosine" algorithm with its iterative propagation variants. While iterative propagation was capable of neatly improving accuracy for niche/difficult judgements, it does so at the expense of robustness, as there are now higher chances of traversing an attack edge, thus sinking into the Sybil sub-region. Note however that, while the ranking of the malicious judgement does increase, the impact of the attack is much smaller than on CF approaches: only in 10% or less cases the malicious judgement reaches the first position for targeted nodes (right), while the change of ranking is almost unperceivable for non-targeted nodes (left).

The addition of γ has the effect of making the recommendations expressed by highly correlated peers more relevant. Our results, shown in Figure 14, match the intuition that this alteration has very little effect on the untargeted attack, where the judgements copied by the attackers are basically chosen at random; the impact is very strong when the attacker is creating Sybil nodes that match the profile of the victim instead. We must point out though that this attack, while effective, does need resources: in this case, we fooled 1% of the honest network in trusting

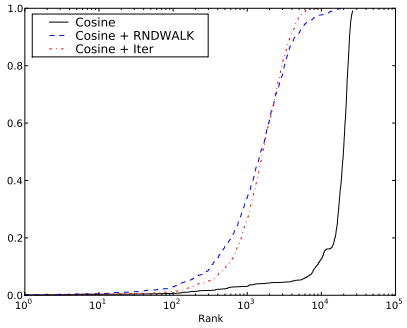


k	n	Percentiles		
		10	50	90
1,000	100	184	14,790	23,178
100	1,000	8,065	19,701	23,994
100	100	8,531	19,827	24,011

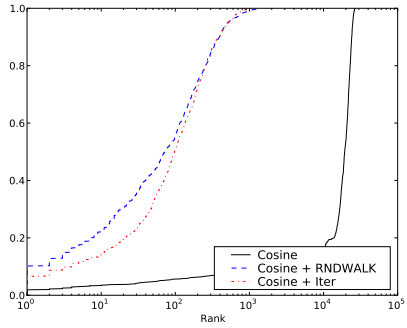


k	n	Percentiles		
		10	50	90
1,000	100	1	8,780	22,941
100	1,000	3,498	19,355	23,855
100	100	6,636	19,307	24,036

Fig. 12: Malicious judgement ranking: PPR + Cosine, with non-targeted (left) and targeted (right) attack. We vary the number k of attack edges and the number n of malicious attackers.



PPR + [Cosine + ...]	Percentiles		
	10	50	90
–	8,531	19,827	24,011
RNDWALK	324	1,524	4,533
Iter	518	1,622	3,784

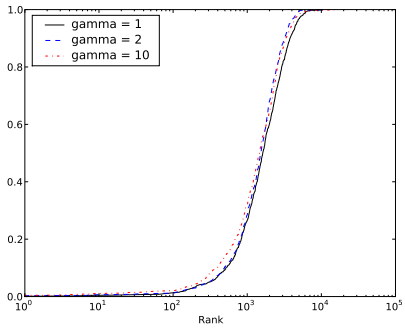


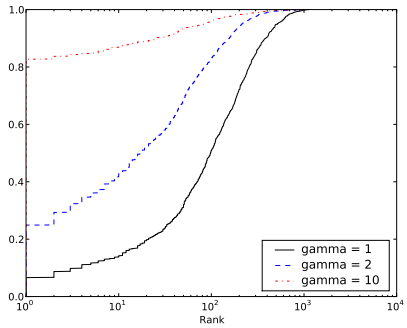
PPR + [Cosine + ...]	Percentiles		
	10	50	90
–	6,636	19,307	24,036
RNDWALK	1	73	395
Iter	4	98	388

Fig. 13: Malicious judgement ranking: iterative trust propagation against non-targeted (left) and targeted (right) attack.

dishonest nodes just to change the recommendations for a single user!

7.2.3 *Lessons Learned.* The experimental assessment of robustness reported above confirms that dependable filtering algorithms do increase resilience against Sybil/ shilling attacks: in fact, unlike traditional collaborative filtering, dependable filtering (in all its variants) is almost completely unaffected by the number of fake nodes in the network, and is still capable of coping well with attacks that require the attacker the (costly) creation of social ties with honest users.





γ	Percentiles				
	10	25	50	75	90
1	518	940	1,622	2,656	3,784
2	493	918	1,527	2,255	3,216
10	397	834	1,462	2,381	3,337

γ	Percentiles				
	10	25	50	75	90
1	4	33	98	219	388
2	1	2	17	66	157
10	1	1	1	1	26

Fig. 14: Malicious judgement ranking: the PPR + [Cosine + Iter + γ] algorithm, with various values of γ (left: non-targeted, right: targeted).

As expected, there exists a trade-off between accuracy and robustness against this latter type of attacks, so that an understanding of the application domain is needed to decide what dependable filtering algorithm is most suited.

8. CONCLUSION

With the advent of Web 2.0 and, most notably, its social networking services, information about users' social ties is becoming widely available. Inspired by approaches that use social networks to defend peer-to-peer systems against Sybil attacks we propose to use such information, together with similarity of tastes, to provide robust and yet accurate recommendations.

We have called this approach dependable filtering, and we proposed a variety of practical realisations. Recommendation quality and attack resistance vary with the particular algorithms adopted, but various results hold. First, while known approaches limit the impact of profile injection attacks to a maximum number of Sybils, we have shown that dependable filtering can provide strong resilience to Sybil/profile injection attacks, even in cases where the number of fake profiles grows indefinitely. Second, we have shown that the accuracy of recommendations produced is at least on par with traditional collaborative filtering algorithms, even when attacks are not in place. Third, the like-mindedness of users that share social links ("homophily") can be exploited to obtain recommendations that in some cases perform better than traditional algorithms based exclusively on judgement similarity. A single best protocol variant does not exist among those proposed, and a tradeoff between accuracy and robustness much be struck, based on the characteristics of the target scenario.

We highlight three possible areas for future work. First, our datasets only contain non-numeric judgements; this made it impossible for us to compare recommendations with a large part of the state of the art in collaborative filtering, and to

implement and validate dependable filtering algorithms that are inspired by those systems. By obtaining datasets containing both social networks and numeric judgements, those comparisons become possible.

A second area of further development regards computational complexity of dependable filtering algorithms, in order to make distributed scalable implementations feasible. The Personalized PageRank is costly to compute, and works that aim to reduce its cost and approximate results, such as the one mentioned in Section 4.1, can be considered. These techniques need to be carefully examined in order to verify whether an attacker could use the approximation of the technique at its own advantage.

A third research area regards the cold-start problem. Users that do not have contacts in a particular social network cannot receive recommendations from dependable filtering algorithms; solutions might require the creation of pre-trusted nodes [Kamvar et al. 2003], or ways of transporting social information between different social networks, as done in the FOAF project [Kruk 2004]. In the case where social relationships have different semantics in different networks (e.g., one social network reflects close trust bindings, while another one only represents weak social connections), links might need to be differentiated depending on their origin.

REFERENCES

Aggarwal, C. C., Wolf, J. L., Wu, K.-L., and Yu, P. S. 1999. Horting hatches an egg: a new graph-theoretic approach to collaborative filtering. In *Proceedings of the 5th ACM SIGKDD international conference on Knowledge discovery and data mining*. 201–212.

Anderson, C. 2006. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion.

Avesani, P., Massa, P., and Tiella, R. 2005. A trust-enhanced recommender system application: Moleskiing. In *Proc. of ACM Symposium on Applied Computing*. ACM, New York, NY, USA, 1589–1593.

Axelrod, R. 1984. *The Evolution of Cooperation*. Basic Books, New York.

Benevenuto, F., Rodrigues, T., Almeida, V., Almeida, J., Zhang, C., and Ross, K. 2008. Identifying video spammers in online social networks. In *Proc. of the 4th International Workshop on Adversarial Information Retrieval on the Web*. ACM, New York, NY, USA, 45–52.

Bhaumik, R., Burke, R., and Mobasher., B. 2007. Effectiveness of crawling attacks against web-based recommender systems. In *Proceedings of AAAI Workshop on Intelligent Techniques for Web Personalization*. 17–26.

Bhaumik, R., Williams, C., Mobasher, B., and Burke, R. 2006. Securing collaborative filtering against malicious attacks through anomaly detection. In *Proceedings of the 4th Workshop on Intelligent Techniques for Web Personalization*.

Bonhard, P. and Sasse, M. A. 2006. 'Knowing me, knowing you' – using profiles and social networking to improve recommender systems. *BT Technology Journal 24,* 3, 84–98.

Burke, R., Mobasher, B., Bhaumik, R., and Williams, C. 2005. Segment-based injection attacks against collaborative filtering recommender systems. In *Proceedings 5th IEEE International Conference on Data Mining*.

Chen, H. and Ng, T. 1999. An algorithmic approach to concept exploration in a large knowledge network (automatic thesaurus consultation): Symbolic branch-and-bound search vs. connectionist Hopfield net activation. *Journal of the American Society for Information Science 46,* 5, 348–369.

Cheng, A. and Friedman, E. 2005. Sybilproof reputation mechanisms. In *Proceedings of 3rd Workshop on Economics of Peer-to-Peer Systems (P2PECON 2005), Philadelphia, PA.*

Chirita, P.-A., Nejdl, W., and Zamfir, C. 2005. Preventing shilling attacks in online recommender systems. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*. 67–74.

Dell'Amico, M. and Capra, L. 2008. SOFIA: Social Filtering for Robust Recommendations. In *Proc. of 2nd Joint iTrust and PST Conferences on Privacy, Trust management and Security*. Trondheim, Norway, 135–150.

Deshpande, M. and Karypis, G. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS) 22,* 1, 177.

Douceur, J. R. 2002. The Sybil attack. In *1st International Workshop on Peer-to-Peer Systems (IPTPS '02), Cambridge, MA, USA*.

Feldman, M., Lai, K., Stoica, I., and Chuang, J. 2004. Robust incentive techniques for peer-to-peer networks. In *Proceedings of ACM Conference on Electronic Commerce (EC'04), New York, NY, USA*.

Fogaras, D., Racz, B., Csalogany, K., and Sarlos, T. 2005. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. *Internet Mathematics 2,* 3.

Friedman, E. J. and Cheng, A. 2006. Manipulability of PageRank under Sybil strategies. In *Proceedings of the 1st Workshop on Networked Systems (NetEcon06), Ann Arbor, MI, USA*.

Gleich, D. and Polito, M. 2006. Approximating personalized pagerank with minimal use of webgraph data. *Internet Mathematics 3,* 3, 257–294.

Golbeck, J. 2005. Computing and applying trust in web-based social networks. Ph.D. thesis, University of Maryland.

Golbeck, J. 2008. Weaving a Web of Trust. *Science*, 1640–1641.

Groh, G. and Ehmig, C. 2007. Recommendations in taste related domains: collaborative filtering vs. social filtering. In *Proceedings of the 2007 international ACM conference on Supporting group work*. 127–136.

Guha, R., Kumar, R., Raghavan, P., and Tomkins, A. 2004. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web (WWW2004), New York, NY, USA*. 403–412.

Gupta, M., Judge, P., and Ammar, M. 2003. A reputation system for peer-to-peer networks. In *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*. 144–152.

Gupta, M., Pathak, A., and Chakrabarti, S. 2008. Fast algorithms for top-k personalized pagerank queries. In *Proceeding of the 17th international conference on World Wide Web*. 1225–1226.

Haveliwala, T., Kamvar, A., and Jeh, G. 2003. An analytical comparison of approaches to personalizing Pagerank. Tech. rep., Stanford InfoLab.

Haveliwala, T. and Kamvar, S. 2003. The second eigenvalue of the google matrix. Tech. Rep. 20/2003, Stanford University.

Haveliwala, T. H. 2003. Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering 15*, 2003.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International Conference on Research and Development in Information Retrieval (SIGIR '99)*. ACM Press, New York, NY, USA, 230–237.

Hu, Y., Koren, Y., and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *ICDM '08: Proceedings of the 2008 8th IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 263–272.

Huang, Z., Chen, H., and Zeng, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems 22,* 1, 116–142.

Huang, Z., Li, X., and Chen, H. 2005. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. 141–142.

Huang, Z. and Zeng, D. 2005. Why does collaborative filtering work? - recommendation model validation and selection by analyzing random bipartite graphs. In *15th Annual Workshop on Information Technologies and Systems*. 33–38.

Huang, Z., Zeng, D. D., and Chen, H. 2007. Analyzing consumer-product graphs: Empirical findings and applications in recommender systems. *Management Science 53,* 7, 1146–1164.

Hurley, N. J., O'Mahony, M. P., and Silvestre, G. C. M. 2007. Attacking recommender systems: A cost-benefit analysis. *IEEE Intelligent Systems 22,* 3, 64–68.

Jeh, G. and Widom, J. 2003. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*. 271–279.

Jösang, A. 1996. The right type of trust for distributed systems. In *NSPW '96: Proceedings of the 1996 workshop on new security paradigms, New York, NY, USA*. ACM Press, 119–131.

Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. 2003. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of WWW 2003*. 640–651.

Kautz, H., Selman, B., and Shah, M. 1997. Referral web: combining social networks and collaborative filtering. *Commun. ACM 40,* 3 (March), 63–65.

Kleinberg, J. M. 1999. Authoritative sources in a hyperlinked environment. *JACM: Journal of the ACM 46*, 604–632.

Kofman, I. and Rajaraman, S. 2010. New video page launches for all users. Official YouTube blog, `http://youtube-global.blogspot.com/2010/03/new-video-page-launches-for-all-users.html`.

Kruk, S. 2004. FOAF-Realm-control your friends' access to the resource. In *FOAF Workshop proceedings*.

Lam, S., Frankowski, D., and Riedl, J. 2006. Do you trust your recommendations? An exploration of security and privacy issues in recommender systems. *Emerging Trends in Information and Communication Security*, 14–29.

Lam, S. K. and Riedl, J. 2004. Shilling recommender systems for fun and profit. In *WWW '04: Proceedings of the 13th international conference on World Wide Web, New York, NY, USA*. ACM Press, 393–402.

Langheinrich, M. 2003. When trust does not compute – the role of trust in ubiquitous computing. Workshop on Privacy at Ubicomp 2003, Seattle, Washington.

Lathia, N., Hailes, S., and Capra, L. 2008a. kNN CF: A Temporal Social Network. In *Proc. of 2nd ACM International Conference on Recommender Systems*. ACM Press, Lausanne, Switzerland.

Lathia, N., Hailes, S., and Capra, L. 2008b. The Effect of Correlation Coefficients on Communities of Recommenders. In *Proc. of 23rd Annual ACM Symposium on Applied Computing - Trust, Recommendations, Evidence and other Collaboration Know-how (TRECK) Track*. Fortaleza, Ceara, Brazil, 2000–2005.

Lempel, R. and Moran, S. 2001. Salsa: the stochastic approach for link-structure analysis. *ACM Trans. Inf. Syst. 19,* 2 (April), 131–160.

Massa, P. and Avesani, P. 2007. Trust-aware recommender systems. In *Proceedings of ACM Recommender Systems Conference, Minneapolis, MN, USA*.

Mehta, B. and Hofmann, T. 2008. A survey of attack-resistant collaborative filtering algorithms. *Bulletin of the Technical Committee on Data Engineering 31,* 2 (June), 14–22.

Mobasher, B., Burke, R., Bhaumik, R., and Sandvig, J. J. 2007. Attacks and remedies in collaborative recommendation. *IEEE Intelligent Systems 22,* 3, 56–63.

Mobasher, B., Burke, R., Bhaumik, R., and Williams, C. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Inter. Tech. 7,* 4, 23.

Mobasher, B., Burke, R., and Sandvig, J. J. 2006. Model-based collaborative filtering as a defense against profile injection attacks. In *Proceedings of the 21st Conference on Artificial Intelligence (AAAI 2006)*.

Nowak, M. A. and Sigmund, K. 1998. Evolution of indirect reciprocity by image scoring. *Nature 393,* 6685, 573–577.

O'Donovan, J. and Smyth, B. 2006. Is trust robust?: an analysis of trust-based recommendation. In *Proceedings of the 11th international conference on Intelligent user interfaces*. 101–108.

O'Mahony, M. P., Hurley, N. J., and Silvestre, G. C. M. 2004. An evaluation of neighbourhood formation on the performance of collaborative filtering. *Artificial Intelligence Review 21,* 3 (June), 215–228.

O'Mahony, M., Hurley, N., and Silvestre, G. 2002. Promoting recommendations: An attack on collaborative filtering. 213–241.

Page, L., Brin, S., Motwani, R., and Winograd, T. 1998. The PageRank citation ranking: Bringing order to the web. Tech. rep., Stanford Digital Library Technologies Project.

Rajaraman, S. 2009. Five stars dominate ratings. Official YouTube blog, `http://youtube-global.blogspot.com/2009/09/five-stars-dominate-ratings.html`.

Ray, S. and Mahanti, A. 2008. Strategies for effective shilling attacks against recommender systems. In *2nd ACM SIGKDD International Workshop on Privacy, Security, and Trust*. 111–125.

Resnick, P. and Sami, R. 2007. The influence limiter: provably manipulation-resistant recommender systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*. ACM, New York, NY, USA, 25–32.

Resnick, P. and Sami, R. 2008. The information cost of manipulation-resistance in recommender systems. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*. ACM, New York, NY, USA, 147–154.

Sandvig, J., Mobasher, B., and Burke, R. 2007a. Impact of relevance measures on the robustness and accuracy of collaborative filtering. 99–108.

Sandvig, J. J., Mobasher, B., and Burke, R. 2007b. Robustness of collaborative recommendation based on association rule mining. In *Proceedings of the ACM conference on Recommender systems*. 105–112.

Sandvig, J. J., Mobasher, B., and Burke, R. 2008. A survey of collaborative recommendation and the robustness of model-based algorithms. *Bulletin of the Technical Committee on Data Engineering 31,* 2 (June), 3–13.

Schollmeier, R. 2001. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Proceedings of the 1st International Conference on Peer-to-Peer Computing*.

Wang, J., de Vries, A., and Reinders, M. 2006. A user-item relevance model for log-based collaborative filtering. In *Advances in Information Retrieval*, M. Lalmas, A. MacFarlane, S. Rüger, A. Tombros, T. Tsikrika, and A. Yavlinsky, Eds. Vol. 3936. Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter 5, 37–48.

Williams, C., Mobasher, B., and Burke, R. 2007. Defending recommender systems: detection of profile injection attacks. *Service Oriented Computing and Applications 1,* 3 (November), 157–170.

Yu, H., Kaminsky, M., Gibbons, P. B., and Flaxman, A. 2006. Sybilguard: defending against sybil attacks via social networks. In *Proceedings of ACM SIGCOMM 2006*. 267–278.

Zhang, S., Chakrabarti, A., Ford, J., and Makedon, F. 2006. Attack detection in time series for recommender systems. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 809–814.

Zheng, R., Provost, F., and Ghose, A. 2007. Social network collaborative filtering. Tech. Rep. CeDER-07-04, New York University. September.

Received December 2008; revised June 2010; accepted July 2010