Spreading Activation Models for Trust Propagation

Cai-Nicolas Ziegler Georg Lausen

Institut für Informatik, Universität Freiburg
Georges-Köhler-Allee, Gebäude 51
79110 Freiburg i.Br., Germany
{cziegler,lausen}@informatik.uni-freiburg.de

Abstract

Semantic Web endeavors have mainly focused on issues pertaining to knowledge representation and ontology design. However, besides understanding information metadata stated by subjects, knowing about their credibility becomes equally crucial. Hence, trust and trust metrics, conceived as computational means to evaluate trust relationships between individuals, come into play. Our major contributions to Semantic Web trust management through this paper are twofold. First, we introduce our classification scheme for trust metrics along various axes and discuss advantages and drawbacks of existing approaches for Semantic Web scenarios. Hereby, we will devise our advocacy for local group trust metrics, guiding us to the second part which presents Appleseed, our novel proposal for local group trust computation. Compelling in its simplicity, Appleseed borrows many ideas from spreading activation models in psychology and relates their concepts to trust evaluation in an intuitive fashion.

1. Introduction

In our world of information overload and global connectivity leveraged through the Web and other types of media, social trust [26] between individuals becomes an invaluable and precious good. Hereby, trust exerts an enormous impact on decisions whether to believe or disbelieve information asserted by other peers. Belief should only be accorded to statements from people we deem trustworthy. However, when supposing huge networks such as the Semantic Web, trust judgements based on personal experience and acquaintanceship become unfeasible. In general, we accord trust, concisely defined by Mui as the "subjective expectation an agent has about another's future behavior based on the history of their encounters" [28], to only small numbers of people. These people, again, trust another limited set of people, and so forth. The network structure emanating from our very person, composed of trust statements linking individuals, constitutes the basis for trusting people we do not know personally. Playing an important role for the conception of Semantic Web trust infrastructure, latter structure has been dubbed "Web of Trust" [11].

We might be tempted to adopt the policy of trusting all those people who are trusted by persons we trust. Trust would thus propagate through the network and become accorded whenever two individuals can reach each other via at least one trust path. However, common sense tells us we should not rely upon this strategy. More complex metrics are needed in order to more sensibly evaluate trust between two persons. Among other features, these trust metrics must take into account social and psychological aspects of trust and suffice criteria of computability and scalability likewise.

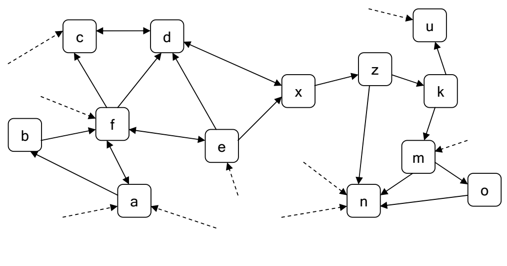

Figure 1. Sample web of trust for agent *a*

Our paper is structured as follows. In order to assess diverse properties of metrics, section 2.1 briefly introduces existing trust metrics and classifies them according to our proposed classification scheme. An investigation of trust metric classes and their fitness for Semantic Web scenarios follows in section 2.2, along with an overview of our asserted trust model. Besides, section 2.2 exposes the urging need for *local group* trust metrics and gives examples of possible application scenarios. Section 3 forms the second part of the paper and explicitly deals with these local group trust metrics. We briefly sketch the well-known Advogato trust metric and introduce our novel Appleseed trust metric in section 3.2. Appleseed constitutes the major contribution of this paper and represents our own approach to local group trust computation. Many of its ideas and concepts borrow from spreading activation models, which simulate human semantic memory. Section 3.3 matches Appleseed and Advogato against each other, discussing advantages and drawbacks of either approach. Furthermore, results of experiments conducted to evaluate the behavior of Appleseed under diverse conditions are illustrated in section 3.4. Eventually, in section 3.5, we indicate possible modifications and give some implementation details, while section 3.6 briefly presents the testbed we have used to base all our experiments and comparisons upon.

2. Trust in Social Networks

Trust represents an invaluable and precious good one should award deliberately. Trust metrics compute quantitative *estimates* of how much trust an agent *a* should accord to its peer *b*, taking into account trust ratings from other persons on the network. These metrics should also act “deliberately”, not overly awarding trust to persons or agents whose trustworthiness is questionable.

2.1. Classification of Trust Metrics

Applications for trust metrics and trust management [4] are rife and not confined to the Semantic Web. First proposals for metrics range back to the early nineties, where trust metrics have been deployed in various projects to support the Public Key Infrastructure [38]. Metrics proposed in [21], [32], [25], and [3] count among the most popular ones for public key authentication and have initiated fruitful discussions. New areas and research fields apart from PKI have come to make trust metrics gain momentum. P2P networks, ubiquitous, mobile computing, and rating systems for online communities, where maintenance of explicit certification authorities is not feasible anymore, have raised the research interest in trust. The whole plethora of available metrics can hereby be defined and characterized along various classification axes. We identify three principal dimensions with distinctive features. These axes are not orthogonal, though, for various features impose restrictions on the feature range of other dimensions. Mind that some of the below mentioned categories have already been defined in prior work. For instance, [13] differentiates between local and global trust, and distinctive features between scalar and group trust metrics are discussed in [20]. However, to our knowledge, no explicit categorization of trust metrics along various axes, supplemented with an analysis of axis interaction, exists. We therefore regard the classification scheme provided below as one major contribution of this paper. Its results are also synthesized in Figure 2.

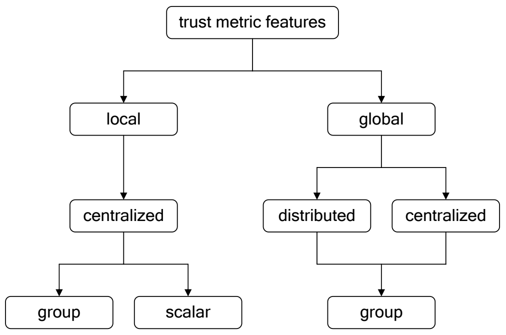

Figure 2. Trust metric classification

2.1.1. Network Perspective. Latter dimension influences semantics assigned to the values computed. Trust metrics may basically be subdivided into ones with *global*, and ones with *local* scope. Global trust metrics take into account all peers and trust links connecting them. Global trust ranks are assigned to an individual based upon complete trust graph information. Many global trust metrics, such as those presented in [17], [13], and [33], borrow their ideas from the renowned PageRank algorithm [29] to compute web page reputation. The basic intuition behind the approach is that nodes should be ranked higher the better the rank of nodes pointing to them. Obviously, latter approach works for trust and page reputation likewise.

Trust metrics with local scope, on the other hand, take into account personal bias. Interestingly, some researchers claim that only local trust metrics are “true” trust metrics, since global ones compute overall reputation rather than personalized trust[1] [28]. Local trust

1 Recall the definition of trust given before, telling that trust is a “subjective expectation”.

metrics take the agent for whom to compute trust as an additional input parameter and are able to operate on *partial* trust graph information. The rationale behind local trust metrics is that persons an agent a trusts may be completely different from the range of individuals that agent b deems trustworthy. Local trust metrics exploit structural information defined by personalized webs of trust. Hereby, the personal web of trust for individual a is given through the set of trust relationships emanating from a and passing through nodes it trusts either directly or indirectly, as well as the set of nodes reachable through latter relationships. Merging all webs of trust engenders the global trust graph. Local trust metrics comprise Levien's Advogato trust metric [22], metrics for modelling the Public Key Infrastructure [3, 25, 32], Golbeck's metrics for Semantic Web trust [11], and Sun Microsystem's Poblano [6]. The latter mentioned work hereby strongly resembles Abdul-Rahman and Hailes [1].

2.1.2. Computation Locus. The second axis refers to the place where trust relationships between individuals are evaluated and quantified. Local[2] or centralized approaches perform all computations in one single machine and hence need to be granted full access to relevant trust information. The trust data itself may hereby be distributed over the network. Most of the before-mentioned metrics count among the class of centralized approaches.

Distributed metrics for computation of trust and reputation, such as those described in [33], [17], and [34], equally deploy the load of computation on every trust node in the network. Upon receiving trust information from its predecessor nodes in the trust graph, an agent merges the data with its own trust assertions and propagates synthesized values to its successor nodes. The entire process of trust computation is necessarily asynchronous and its convergence depends on the eagerness or laziness of nodes to propagate information. Another characteristic feature of distributed trust metrics refers to the fact that they are inherently global. Though the individual computation load is decreased with respect to centralized computation approaches, nodes need to store trust information about *any other* node in the system.

2.1.3. Link Evaluation. Latter axis distinguishes scalar and group trust metrics. According to Levien [20], scalar metrics analyze trust assertions independently, while group trust metrics evaluate groups of assertions "in tandem". PageRank [29] and related approaches count among global group trust metrics, for the reputation of one page depends on the ranks of referring pages, thus entailing parallel evaluation of relevant nodes thanks to mutual dependencies. Advogato [22] represents an example for local group trust metrics. Most other trust metrics count among the category of scalar ones, tracking trust paths from sources to targets and not performing parallel evaluations of groups of trust assertions. Hence, another basic difference between scalar and group trust metrics refers to their functional design. In general, scalar metrics compute trust between two given individuals a and b taken from the set of agents V.

On the other hand, group trust metrics generally compute trust ranks for *sets* of individuals from V. Hereby, global group trust metrics assign trust ranks for every $a \in V$, while local ones may also return ranked subsets of V. Note that complete trust graph information is only important for global group trust metrics, but not for local ones. Informally, local group trust metrics may be defined as metrics to compute neighborhoods of trusted peers for an individual a. As input parameters, these trust metrics take an individual $a \in V$ for which to compute the set of peers it should trust, as well as an amount of trust the latter wants to share among the most trustworthy agents. For instance, in [22], the amount of trust is said to correspond to the number of agents that a wants to trust. The output is hence given by a trusted subset of V.

Note that scalar trust metrics are inherently local, while group trust metrics do not impose any restrictions on features for other axes.

2.2. Semantic Web Trust

Most presented metrics and trust models have been proposed for scenarios other than the Semantic Web. In fact, research in trust infrastructure and metrics for latter network of metadata still has to come of age and gain momentum. Before discussing specific requirements and fitness properties of trust metrics along those axes proposed before, we need to define one common trust model which to rely upon for the Semantic Web. Some steps towards one such common model have already been taken and incorporated into the FOAF [7] project. FOAF is an abbreviation for "Friend of a Friend" and aims at enriching personal homepages with machine-readable content encoded in RDF statements. Besides various other information, these publicly accessible pages allow their owners to nominate all individuals part of the FOAF universe

2 Mind that in this context, "local" refers to the place of *computation* and not the kind of data evaluation.

they know, thus weaving a "web of acquaintances" [11]. Golbeck has extended the FOAF schema to also contain *trust* assertions with values ranging from 1 to 9, where 1 denotes complete distrust and 9 absolute trust towards the individual for which the assertion has been issued [11]. The model that we adopt is quite similar to FOAF and its extensions, but only captures the notion of trust and lack of trust, instead of trust and distrust. Note that zero trust and distrust are not the same [24] and may hence not be intermingled. Explicit modelling of distrust has some serious implications for trust metrics and still needs to become subject of intense study. First steps towards the implementation of distrust have been taken by Jøsang et al. [15] and Guha [13].

2.2.1. Trust Model. In this section, we present the constituents of our model for Semantic Web trust infrastructure. As it is the case for FOAF, we assume that all trust information is publicly accessible for any agent in the system through machine-readable personal homepages distributed over the network. This assumption may yield privacy concerns and will be discussed and justified later.

- **Agent set $V = \{a_1, \ldots, a_n\}$.** Similar to the FOAF approach, we assume agents $a \in V$ to be represented and uniquely identified by the URI of their machine-readable personal homepage.
- **Partial trust function set $T = \{W_{a_1}, \ldots, W_{a_n}\}$.** Every agent a is associated with one partial trust function $W_a : V \to [0,1]^{\perp}$, which corresponds to the set of trust assertions that a has stated on his machine-readable homepage. In most cases, these functions will be very sparse as the number of individuals for which an agent is able to assign explicit trust ratings is much smaller than the total number n of agents on the Semantic Web:

$$W_{a_i}(a_j) = \begin{cases} p, & \text{if trust}(a_i, a_j) = p \\ \perp, & \text{if no rating for } a_j \text{ from } a_i \end{cases}$$

Note that the higher the value of $W_{a_i}(a_j)$, the more trustworthy a_i deems a_j. Conversely, $W_{a_i}(a_j) = 0$ means that a_i considers a_j to be not trustworthy at all. The assignment of trust through continuous values between 0 and 1 and their adopted semantics is in perfect accordance with [23], where possible stratifications of trust values are proposed. Our trust model defines one directed trust graph with nodes being represented by agents $a \in V$ and directed edges from nodes a_i to nodes a_j being trust statements with weight $W_{a_i}(a_j)$.

For convenience, we furthermore introduce partial function $W : V \times V \to [0,1]^{\perp}$ which we define as the union of all partial functions $W_a \in T$.

2.2.2. Trust Metrics for the Semantic Web. Trust and reputation ranking metrics have primarily been used for public key certification [31, 32, 21, 25, 3], rating and reputation systems part of online communities [13, 22, 20], P2P networks [17, 34, 19, 18, 2], and also mobile computing fields [8]. Each of these scenarios favors different trust metrics. For instance, reputation systems for online communities tend to make use of centralized trust servers that compute global trust values for all users in the system [13]. On the other hand, P2P networks with moderate size rely upon distributed approaches that are in most cases based upon PageRank [17, 34].

The Semantic Web, however, is expected to be made up of millions of nodes a representing agents. The fitness of *distributed* approaches to trust metric computation, such as [33] and [17], is hence limited because of various reasons:

- **Trust data storage.** Each agent a needs to store trust information about any other agent b on the Semantic Web. Agent a uses this information in order to merge it with own trust beliefs and propagates the synthesized information to trusted agents. Even though we might expect the size of the Semantic Web to be several orders of magnitude smaller than the traditional Web, the number of agents which to keep trust information for will still exceed storage capabilities of "normal" agents.
- **Convergence.** The structure of the Semantic Web is diffuse and not subject to some higher ordering principle or hierarchy. Furthermore, the process of trust propagation is necessarily asynchronous. As the Semantic Web is huge in size with possibly numerous antagonist or idle agents, convergence of trust values might take a very long time.

The huge advantage of distributed approaches, on the other hand, is the immediate availability of computed trust information for any other agent in the system as well as the fact that agents have to disclose their trust assertions only to peers they trust [33]. For instance, suppose that a declares its trust in b to be 0.1, which is very low. Hence, a might want b not to know about that fact. As distributed metrics only propagate synthesized trust values from nodes to successor nodes in the trust graph, a would not have to disclose its trust statements to b.

As it comes to centralized, i.e., locally computed, metrics, full trust information access is required for agents computing trust. Hence, online communities based on trust require their users to disclose all trust information to the community server, but not necessarily to other peers [13]. Privacy is thus maintained. On the Semantic Web and in the area of ubiquitous and mobile computing, however, it is not only some central authority which computes trust. Any agent might want to do so. Our own trust model, as well as trust models proposed in [11], [8], and [1], are hence based upon the assumption of publicly available trust information. Though privacy concerns may persist, this assumption is vital due to the mentioned deficiencies of distributed computation models. Moreover, centralized *global* metrics, such as depicted in [13] and [29], also fail to fit the requirements imposed by the Semantic Web: due to the huge number of agents issuing trust statements, only dedicated server clusters could be able to manage the whole bulk of trust relationships. For small agents and applications roaming the Semantic Web, global trust computation is not feasible.

The traditional as well as the Semantic Web bear significant traits of small world networks [11]. Small worlds theory has been investigated extensively by Stanley Milgram, social psychologist at Harvard University. His hypothesis, commonly referred to as "six degrees of separation", states that members of any large social network are connected to each other through short chains of intermediate acquaintances [12]. Relating his research results to trust on the Semantic Web, we come to conclude that average trust path lengths between any two individuals are small. Hence, locally computed *local* trust metrics considering trust paths from trust sources to trust targets, such as the ones proposed for PKI [31, 32, 21, 25, 3], may be expected to suitably lend themselves to the Semantic Web. In contrast to global metrics, no clustering of massive CPU power is required to compute trust.

Besides centrally computed *scalar* trust metrics taking into account personal bias, we advocate *local group* trust metrics for the Semantic Web. These metrics bear several welcome properties with respect to computability and complexity, which may be summarized as follows:

- **Partial trust graph exploration.** Global metrics require a priori full knowledge of the entire trust network. Distributed metrics store trust values for all agents in the system, thus implying massive data storage demands. On the other hand, when computing trusted neighborhoods, the trust network only needs to be explored partially: originating from the trust source, one only follows those trust edges that seem promising, i.e., bearing high trust weights, and which are not too far away from the trust source. Inspection of personal, machine-readable homepages is thus performed in a Just-In-Time fashion. Hence, prefetching bulk trust information is not required.
- **Computational scalability.** Tightly intertwined with partial trust graph exploration is computational complexity. Local group trust metrics scale well to any social network size, as only tiny subsets of relatively constant size[3] are visited. This is not the case for global trust metrics.

By the time of this writing, local group trust metrics are subject to comparatively sparse research and none, to our knowledge, have been proposed for the Semantic Web. However, we believe that local group trust metrics will play an important role for trust communities on the Semantic Web. Application scenarios for group trust are rife. In order to not go beyond the scope of this article, we will give just one detailed example dealing with trust in metadata statements:

The Semantic Web basically consists of metadata assertions that machines can understand thanks to ontology sharing. However, since the number of agents able to publish statements is vast, credibility in those statements should be limited. The issue of trust in Semantic Web content has already been addressed in [10]. Herein, the authors propose a centralized system which allows issuing statements and analyzing their reliability and credibility. Complementary to this work by Gil and Ratnakar, the W3C Annotea Project intends to provide an infrastructure for assigning annotations to statements [16]. These statements could also include statements about the credibility of certain metadata. Supposing such an environment and supposing an agent a who wants to reason about the credibility of an assertion s found on the Semantic Web, local group trust metrics could play an important role in its quest: not being able to judge the credibility of s on its own, a could refer to its personal web of trust and compute its n most trusted peers. The latter trust neighborhood is now taking part in an opinion poll where a wants to know about the credibility its trusted peers assign to s. Technically, this could be achieved by searching Annotea servers for statements by a's peers about s. The eventual decision whether to believe s or not could then be made by averaging the credibility ratings of its trusted peers. Similar mod-

3 Supposing identical parameterizations for the metrics in use, as well as similar network structures.

els with distributed reputation systems based on trust have been proposed in [19].

3. Local Group Trust Metrics

Local group trust metrics, in their function as means to compute trust neighborhoods, have not been subject to mainstream research until now. Actually, significant research has been limited to the work done by Levien [20, 21], having conceived the Advogato group trust metric. This section provides an overview of Advogato and introduces our own Appleseed trust metric, eventually comparing both approaches.

3.1. Outline of Advogato Maxflow

The Advogato maximum flow trust metric has been proposed by Levien [22] in order to discover which users are trusted by members of an online community and which are not. Hereby, trust is computed by a centralized community server and considered relative to a seed of users enjoying supreme trust. However, the metric is not only applicable to community servers, but also arbitrary agents which may compute own trusted peers and not for the whole community they belong to. In this case, the agent itself constitutes the singleton trust seed. The following paragraphs briefly introduce basic concepts. For more detailed information, refer to [22], [21], and [20].

3.1.1. Trust Computation Steps. Local group trust metrics compute sets of agents trusted by those being part of the trust seed. In case of Advogato, its input is given by an integer number n, which is supposed to be equal to the number of members to trust [22], as well as the trust seed s, being a subset of the entire set of users V. The output is a characteristic function that maps each member to a boolean value indicating trustworthiness:

$$\text{Trust}_M : 2^V \times \mathbb{N}_0^+ \rightarrow (V \rightarrow \{\text{true}, \text{false}\})$$

The trust model underlying Advogato does *not* provide support for weighted trust relationships in its original version.[4] Hence, trust edges extending from individual x to y express blind trust of x in y. Metrics for PKI maintenance suppose similar models. Maximum integer network flow computation [9] has been investigated by Reiter and Stubblebine [32, 31] in order to make trust metrics more reliable. Levien has adopted and extended this approach for group trust in his Advogato metric:

Capacities $C_V : V \rightarrow \mathbb{N}$ are assigned to every community member $x \in V$ based upon the shortest-path distance from the seed to x. Hereby, the capacity of the seed itself is given by the input parameter n mentioned before, whereas the capacity of each successive distance level is equal to the capacity of the previous level l divided by the average outdegree of trust edges $e \in E$ extending from l. The trust graph obtained hence contains one single source, which is the set of seed nodes considered one single “virtual” node, and multiple sinks, i.e., all nodes other than those defining the seed. Capacities $C_V(x)$ are constraining nodes. In order to apply Ford-Fulkerson maximum integer network flow [9], the underlying problem has to be formulated as single-source/single-sink, with capacities $C_E : E \rightarrow \mathbb{N}$ constraining edges instead of nodes. Hence, Algorithm 1 is applied to the old directed graph $G = (V, E, C_V)$, resulting in a new graph structure $G' = (V', E', C_{E'})$.

```
function transform ($G = (V, E, C_V)$) {
  set $E' \leftarrow \emptyset$, $V' \leftarrow \emptyset$;
  for all $x \in V$ do
    add node $x^+$ to $V'$;
    add node $x^-$ to $V'$;
    if $C_V(x) \geq 1$ then
      add edge $(x^-, x^+)$ to $E'$;
      set $C_{E'}(x^-, x^+) \leftarrow C_V(x) - 1$;
      for all $(x, y) \in E$ do
        add edge $(x^+, y^-)$ to $E'$;
        set $C_{E'}(x^+, y^-) \leftarrow \infty$;
      end do
      add edge $(x^-, \text{supersink})$ to $E'$;
      set $C_{E'}(x^-, \text{supersink}) \leftarrow 1$;
    end if
  end do
  return $G' = (V', E', C_{E'})$;
}
```

Algorithm 1. Trust graph conversion

Conversion is followed by simple integer maximum network flow computation from the trust seed to the super-sink. Eventually, trusted agents x are exactly those peers for which there is flow from “negative” nodes x^- to the super-sink. An additional constraint needs to be introduced, requiring flow from x^- to the super-sink whenever there is flow from x^- to x^+. Latter constraint assures that node x does not only

4 Though various levels of peer certification exist, their proposed interpretation does not correspond to weighted trust relationships.

serve as an intermediate for the flow to pass through, but is actually added to the list of trusted agents when reached by network flow. However, the standard implementation of Ford-Fulkerson traces shortest paths to the sink first [9]. Therefore, respective constraint is satisfied implicitly already.

3.1.2. Attack-Resistance Properties. Advogato has been designed with resistance against massive attacks from malicious agents outside of the community in mind. Therefore, an upper bound for the number of "bad" peers chosen by the metric is provided in [22], along with an informal security proof to underpin its fitness. Resistance against malevolent users trying to break into the community may already be observed in the example depicted by Figure 1, supposing n to be "bad": though agent n is trusted by numerous persons, it is deemed less trustworthy than, for instance, x. However, while there are fewer agents trusting x, these agents enjoy higher trust reputation[5] than the numerous persons trusting n. Hence, it is not just the *number* of agents trusting an individual i, but also the trust *reputation* of these agents that exerts an impact on the trust assigned to i. PageRank [29] works in a similar fashion and has been claimed to possess similar properties of attack-resistance like the Advogato trust metric [20]. In order to make the concept of attack-resistance more tangible, Levien proposes the "bottleneck property" as common feature of attack-resistant trust metrics. Informally, respective property states that the "trust quantity accorded to an edge $s \rightarrow t$ is not significantly affected by changes to the successors of t" [20]. Moreover, attack-resistance features of various trust metrics are discussed in detail in [21] and [36].

3.2. Appleseed Trust Metric

The Appleseed trust metric constitutes the main contribution of this paper and is our novel proposal for local group trust metrics. In contrast to Advogato, being inspired by maximum network flow computation, the basic intuition of Appleseed is motivated by spreading activation strategies. Spreading activation models have first been proposed by Quillian [30] in order to simulate human comprehension through semantic memory. They are commonly described as "models of retrieval from long-term memory in which activation subdivides among paths emanating from an activated mental representation" [35]. By the time of this writing, the seminal work of Quillian has been ported to a whole plethora of other disciplines, such as latent semantic indexing and text illustration. As an example, we will briefly introduce the spreading activation approach adopted in [5] for semantic search in contextual network graphs in order to then relate Appleseed to their work.

5 With respect to seed node a.

3.2.1. Searches in Contextual Network Graphs. The graph model underlying search strategies in contextual network graphs is almost identical in structure to the one presented in section 2.2.1, i.e., edges $(x, y) \in E \subseteq V \times V$ connecting nodes $x, y \in V$. Edges are assigned continuous weights through $W : E \rightarrow [0, 1]$. Source node s to start the search from is activated through an injection of energy e, which is then propagated to other nodes along edges according to some set of simple rules: all energy is fully divided among successor nodes with respect to their normalized local edge weight, i.e., the higher the weight of an edge $(x, y) \in E$, the higher the portion of energy that flows along latter edge. Furthermore, supposing average outdegrees greater than one, the closer node x to the injection source s, and the more paths leading from s to x, the higher the amount of energy flowing into x in general. To eliminate endless, marginal and negligible flow, energy streaming into node x must exceed threshold T in order to not run dry. The described approach is captured formally by Algorithm 2, which propagates energy recursively.

```
procedure energize ($e \in \mathbb{R}_0^+, s \in V$) {
  energy($s$) $\leftarrow$ energy($s$) $+ e$;
  $e' \leftarrow e \ / \ \sum_{(s,n) \in E} W(s, n)$;
  if $e > T$ then
    $\forall (s, n) \in E$ : energize ($e' \cdot W(s, n), n$);
  end if
}
```

Algorithm 2. Recursive energy propagation

3.2.2. Trust Propagation. Algorithm 2 shows the basic intuition behind spreading activation models. In order to tailor these models to trust computation, later to become the Appleseed trust metric, serious adaptations are necessary. For instance, procedure energize(e, s) registers *all* energy e that has passed through node x, accumulated in energy(x). Hence, energy(x) represents the *rank* of x. Higher values indicate higher node rank. However, at the same time, all energy contributing to the rank of x is passed *without loss* to its successor nodes. Interpreting energy ranks as trust ranks thus implies numerous issues of semantic consistency as well as computability. Consider the graph depicted on the left-hand side of Figure 3. Applying spreading activation according to [5], trust ranks of nodes b and d will be identical. However, common sense tells us that d should be accorded *less* trust than b, since its shortest-path distance to the trust seed is higher. Trust decay is commonly agreed upon [13, 15], for people tend to trust individuals trusted by own friends more than individuals trusted only by friends of friends. The right-hand side of Figure 3 entails even more serious implications. All energy, or trust[6], respectively, distributed along edge (a, b) becomes trapped in a cycle and will never be accorded to any other nodes but those being part of latter cycle, i.e., b, c, and d. Latter nodes will eventually acquire infinite trust rank. Obviously, the bottleneck property [20] does not hold. Similar issues occur with simplified versions of PageRank [29], where cycles accumulating infinite rank are dubbed "rank sinks".

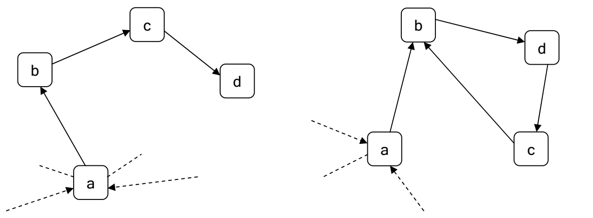

Figure 3. Node chains and rank sinks

3.2.3. Spreading Factor. We handle both issues, i.e., trust decay in node chains and elimination of rank sinks, by tailoring the algorithm to rely upon our global spreading factor d. Hereby, let $\text{in}(x)$ denote the energy influx into node x. Respective parameter d then denotes the portion of energy $d \cdot \text{in}(x)$ that latter node distributes among successors, while retaining $(1 - d) \cdot \text{in}(x)$ for itself. For instance, suppose $d = 0.85$ and energy quantity $\text{in}(x) = 5.0$ flowing into node x. Then, the total energy distributed to successor nodes amounts to 4.25, while energy rank $\text{energy}(x)$ of x increases by 0.75. Special treatment is necessary for nodes with zero outdegree. For simplicity, we assume all nodes to have an outdegree of at least one, which makes perfect sense, as will be shown later.

The spreading factor concept is very intuitive and, in fact, very close to real models of energy spreading through networks. Observe that the overall amount of energy in the network, after initial activation in^0, does not change over time. More formally, suppose that $\text{energy}(n) = 0$ for all $n \in V$ before injection in^0 into source s. Then the following equation holds in every computation step of our modified spreading algorithm, incorporating the concept of spreading factor d:

$$\sum_{x \in V} \text{energy}(x) = \text{in}^0$$

Spreading factor d may also be seen as the ratio between direct trust in x and trust in the ability of x to recommend others as trustworthy peers. For instance, Beth et al. [3] and Maurer [25] explicitly differentiate between *direct* trust edges and *recommendation* edges. We generally assume $d = 0.85$, though other values may also seem reasonable. For instance, having $d \leq 0.5$ allows agents to keep most of the trust they are granted for themselves and only pass small portions of trust accorded to their peers. Observe that low values of d favor trust proximity to the source of trust injection, while high values allow trust to also reach nodes which are further away. Furthermore, the introduction of spreading factor d is crucial to making Appleseed retain Levien's bottleneck property, as will be shown in later sections.

3.2.4. Rank Normalization. Algorithm 2 makes use of edge weight normalization, i.e., the quantity $e_{x \to y}$ of energy distributed along (x, y) from x to successor node y depends on its *relative* weight, i.e., $W(x, y)$ compared to the sum of weights of all outgoing edges of x:

$$e_{x \to y} = d \cdot \text{in}(x) \cdot \frac{W(x, y)}{\sum_{(x,s) \in E} W(x, s)}$$

Normalization is common practice to many trust metrics, among those PageRank [29], EigenTrust [17], and AORank [13]. However, while normalized reputation or trust seems reasonable for models with plain, non-weighted edges, serious interferences occur when edges are weighted, as is the case for our trust model adopted in section 2.2.1.

For instance, refer to the left-hand side of Figure 4 for unwanted effects: the amount of energy that node a accords to successors b and d, i.e., $e_{a \to b}$ and $e_{a \to d}$, respectively, are identical in value. Note that b has issued only *one* trust statement $W(b, d) = 0.25$, telling that its trust in c is rather weak. On the other hand, d assigns *full* trust to individuals e, f, and g. Nevertheless, the overall trust rank for d will be much higher than for any successor of d, for c is accorded $e_{a \to b} \cdot d$, while e, f, and g only obtain $e_{a \to d} \cdot d \cdot 1/3$ each. Hence, c will be trusted three times as much as e, f, and g, which is not reasonable at all.

6 The terms "energy" and "trust" are used interchangeably in this context.

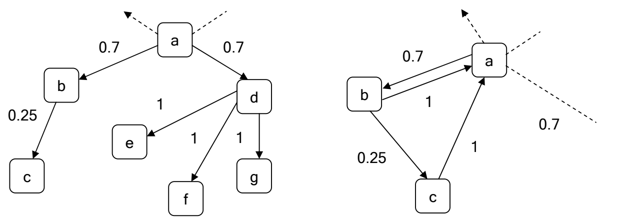

Figure 4. Issues with trust normalization

3.2.5. Backward Trust Propagation. The above issue has already been discussed in [17], but no solution has been proposed therein, arguing that "substantially good results" have been achieved despite the drawbacks. We propose to alleviate the problem by making use of backward propagation of trust to the source: when metric computation takes place, additional "virtual" edges (x, s) from every node $x \in V \setminus \{s\}$ to the trust source s are created. These edges are assigned full trust $W(x, s) = 1$. Existing backward links (x, s), along with their weights, are "overwritten". Intuitively, every node is supposed to blindly trust the trust source s, see Figure 4. Impacts of adding backward propagation links are threefold:

- **Mitigating relative trust.** Again, we refer to the left hand graph in Figure 4. Trust distribution in the underlying case becomes much fairer through backward propagation links, for c now only obtains $e_{a \to b} \cdot d \cdot (0.25/(1 + 0.25))$ from source s, while e, f, and g are accorded $e_{a \to d} \cdot d \cdot (1/4)$ each. Hence, trust ranks of both e, f, and g amount to 1.25 times the trust assigned to c.

- **Avoidance of dead ends.** Dead ends, i.e., nodes x with zero outdegree, require special treatment in our computation scheme. Two distinct approaches may be adopted. First, the portion of incoming trust $d \cdot \text{in}(x)$ supposed to be passed to successor nodes is completely discarded, which contradicts our constraint of no energy leaving the system. Second, instead of retaining $(1 - d) \cdot \text{in}(x)$ of incoming trust, x keeps *all* trust for itself. Latter approach is also not sensible as it encourages users to not issue trust statements for their peers. Luckily, with backward propagation of trust, all nodes are implicitly linked to the trust source s, so that there are no more dead ends to consider.

- **Favoring trust proximity.** Backward links to the trust source s are favorable for nodes close to the source, as their eventual trust rank will increase. On the other hand, nodes further away from s are penalized.

Overly rewarding nodes close to the source is not beyond dispute and may pose some issues. In fact, it represents the tradeoff we have to pay for both welcome aspects of backward propagation.

3.2.6. Nonlinear Trust Normalization. In addition to backward propagation, an integral part of Appleseed, we propose supplementary measures to decrease the negative impact of trust distribution based on relative weights. Situations where nodes x with poor ratings from y are awarded high overall trust ranks, thanks to the low outdegree of y, have to be avoided. Taking the squares of local trust weights provides an appropriate solution:

$$e_{x \to y} = d \cdot \text{in}(x) \cdot \frac{W(x, y)^2}{\sum_{(x,s) \in E} W(x, s)^2}$$

As an example, refer to node b in Figure 4. With squared normalization, the total amount of energy flowing backward to source a increases, while the amount of energy flowing to poorly trusted node c decreases significantly. Accorded trust quantities $e_{b \to a}$ and $e_{b \to c}$ amount to $d \cdot \text{in}(b) \cdot (1/1.0625)$ and $d \cdot \text{in}(b) \cdot (0.0625/1.0625)$, respectively. More serious penalization of poor trust ratings can be achieved by selecting powers above two.

3.2.7. Algorithm Outline. Having identified modifications to apply to spreading activation models in order to tailor them for local group trust metrics, we are now able to formulate the core algorithm of Appleseed. Input and output are characterized as follows:

$$\text{Trust}_A : V \times \mathbb{R}_0^+ \times [0, 1] \times \mathbb{R}^+ \to (\text{trust} : V \to \mathbb{R}_0^+)$$

The first input parameter specifies trust seed s, the second trust injection e, parameter three identifies spreading factor $d \in [0, 1]$, and the fourth argument binds accuracy threshold T_c, which serves as one of two convergence criteria. Similar to Advogato, the output is an assignment function of trust with domain V. However, Appleseed allows *rankings* of agents with respect to trust accorded. Advogato, on the other hand, only assigns boolean values indicating presence or absence of trust.

Appleseed works with *partial* trust graph information. Nodes are accessed only when needed, i.e., when reached by energy flow. Trust ranks trust(x), which correspond to energy(x) in Algorithm 2, are initialized to 0. Any unknown node u hence obtains trust$(u) = 0$. Likewise, virtual trust edges for backward propagation from node x to the source are added the moment that x is discovered. In every iteration, for those nodes x reached by flow, the amount of incoming trust is computed as follows:

$$\text{in}(x) = d \cdot \sum_{(p,x) \in E} \left(\text{in}(p) \cdot \frac{W(p,x)}{\sum_{(p,s) \in E} W(p,s)} \right)$$

Incoming flow for x is hence determined by all flow that predecessors p distribute along edges (p,x). Note that the above equation makes use of linear normalization of relative trust weights. Replacement of linear by nonlinear normalization according to section 3.2.6 is straight-forward, though. The trust rank of x is updated as follows:

$$\text{trust}(x) \leftarrow \text{trust}(x) + (1-d) \cdot \text{in}(x)$$

However, trust networks generally contain cycles and thus allow no topological sorting of nodes. Hence, the computation of $\text{in}(x)$ for reachable $x \in V$ is inherently recursive. Several iterations for all nodes are required in order to make computed information converge towards the least fixpoint. We give two criterions that have to be satisfied for convergence, relying upon accuracy threshold T_c briefly introduced before.

Definition 1 (Termination) Suppose that $V_i \subseteq V$ represents the set of nodes that have been discovered until step i, and $\text{trust}_i(x)$ the current trust ranks for all $x \in V$. Then the algorithm terminates when both conditions are satisfied after step i:

(a) $V_i = V_{i-1}$

(b) $\forall x \in V_i : \text{trust}_i(x) - \text{trust}_{i-1}(x) \leq T_c$

Informally, Appleseed terminates when no new nodes have been discovered and when changes of trust ranks with respect to prior iteration $i-1$ are not greater than accuracy threshold T_c.

3.3. Comparison of Advogato and Appleseed

Advogato and Appleseed both constitute implementations of local group trust metrics. Advogato has already proven its efficiency in practical usage scenarios such as the Advogato online community, though lacking quantitative fitness information. Its success is mainly measured by indirect feedback, such as the amount of spam messages posted on Advogato, which has been claimed to be rather low. In order to evaluate the fitness of Appleseed as an appropriate approach to group trust computation, we intend to relate our novel approach to Advogato for comparison:

```
function Trust_A ($s \in V$, $\text{in}^0 \in \mathbb{R}_0^+$, $d \in [0,1]$, $T_c \in \mathbb{R}^+$) {
  set $\text{in}_0(s) \leftarrow \text{in}^0$, $\text{trust}_0(s) \leftarrow 0$, $i \leftarrow 0$;
  set $V_0 \leftarrow \{s\}$;
  repeat
    set $i \leftarrow i + 1$;
    set $V_i \leftarrow V_{i-1}$;
    for all $x \in V_{i-1}$ do
      set $\text{trust}_i(x) \leftarrow \text{trust}_{i-1}(x) + (1-d) \cdot \text{in}_{i-1}(x)$;
      set $\text{in}_i(x) \leftarrow 0$;
      for all $(x,u) \in E$ do
        if $u \notin V_i$ then
          set $V_i \leftarrow V_i \cup \{u\}$;
          set $\text{trust}_i(u) \leftarrow 0$, $\text{in}_i(u) \leftarrow 0$;
          add edge $(u,s)$, set $W(u,s) \leftarrow 1$;
        end if
        set $w \leftarrow W(x,u) \;/\; \sum_{(x,u') \in E} W(x,u')$;
        set $\text{in}_i(u) \leftarrow \text{in}_i(u) + d \cdot \text{in}_{i-1}(x) \cdot w$;
      end do
    end do
    set $m = \max_{y \in V_i} \{\text{trust}_i(y) - \text{trust}_{i-1}(y)\}$;
  until $(V_i \backslash V_{i-1} = \emptyset \wedge m \leq T_c)$
  return (trust : $\{(x, \text{trust}_i(x)) \mid x \in V_i\}$);
}
```

Algorithm 3. Appleseed trust metric

- **Attack-resistance.** Latter property defines the behavior of trust metrics in case of malicious nodes trying to invade into the system. For evaluation of attack-resistance capabilities, we have briefly introduced the "bottleneck property" in section 3.1.2, which holds for Advogato. In order to recapitulate, suppose that s and t are nodes and connected through trust edge (s,t). Node s is assumed good, while t is an attacking agent trying to make good nodes trust malevolent ones. In case the bottleneck property holds, manipulation "on the part of bad nodes does not affect the trust value" [20]. Clearly, Appleseed satisfies the bottleneck property, for nodes cannot raise their impact by modifying the structure of trust statements they issue. Bear in mind that the amount of trust accorded to agent t *only* depends on its predecessors and does not increase when t adds more nodes. Both spreading factor d and normalization of trust statements ensure Appleseed to become attack-resistant like Advogato.

- **Trust weight normalization.** We have indicated before that issuing multiple trust statements dilutes trust accorded to successors. According to Guha [13], this does not comply with real world observations, where statements of trust "do not decrease in value when the user trusts one more person [...]". The malady that Appleseed suffers from is common to many trust metrics, most notably those based upon finding principal eigenvectors [29, 17, 33]. On the other hand, the approach pursued by Advogato does not penalize trust relationships asserted by eager trust dispensers, for node capacities do not depend on local information. Remember that capacities of nodes pertaining to level l are assigned based on the capacity of level $l-1$ as well as the *overall* outdegree of nodes part of latter level. Hence, Advogato encourages agents issuing numerous trust statements, while Appleseed penalizes abundant trust certificates.

- **Deterministic trust computation.** Appleseed is deterministic with respect to the assignment of trust rank to agents. Hence, for any arbitrary trust graph $G = (V, E, W)$ and for every node $x \in V$, linear equations allow to characterize the amount of trust assigned to x, as well as the quantity that x accords to its successor nodes. Advogato, however, is non-deterministic. Though the *number* of trusted agents, and therefore the computed maximum flow size, is determined for given input parameters, the set of agents itself is not. Changing the order in which trust assertions are issued may yield different results. For example, suppose $C_V(s) = 1$ holds for trust seed s. Furthermore, assume s has issued trust certificates for two agents, b and c. The actual choice between b or c as trustworthy peer with maximum flow only depends on the order in which nodes are accessed.

- **Model and output type.** Basically, Advogato supports non-weighted trust statements only. Appleseed is more versatile thanks to its trust model based on weighted trust certificates. In addition, Advogato returns one set of trusted peers, whereas Appleseed assigns *ranks* to agents. These ranks allow to select most trustworthy agents first and relate them to each other with respect to their accorded rank. Hereby, the definition of thresholds for trustworthiness is left to the user who can thus tailor relevant parameters to fit different application scenarios. For instance, raising the application-dependent threshold for selection of trustworthy peers, which may be either an absolute or relative value, allows for enlarging the neighborhood of trusted peers. Appleseed is hence more adaptive and flexible than Advogato.

3.4. Parameterization and Experiments

Appleseed allows numerous parameterizations of input variables. Discussions of parameter instantiations and caveats thus constitute indispensable complements to our contribution. Moreover, we provide experimental results exposing observed effects of parameter tuning. Note that all experiments have been conducted on data obtained from "real" social networks: several web crawling tools were written to mine the Advogato community web site and extract trust assertions stated by its more than 8,000 members. Hereafter, we converted all trust data to our trust model proposed in section 2.2.1. Notice that the Advogato community server supports four different levels of peer certification, namely "Observer", "Apprentice", "Journeyer", and "Master". We mapped these qualitative certification levels to quantitative ones, assigning $W(x, y) = 0.25$ for x certifying y as "Observer", $W(x, y) = 0.5$ for an "Apprentice", and so forth. The Advogato community grows rapidly and our crawler extracted $3,224,101$ trust assertions. Heavy preprocessing and data cleansing was inevitable, eliminating reflexive trust statements $W(x, x)$ and shrinking trust certificates to reasonable sizes. Note that some eager Advogato members have issued more than two thousand trust statements, yielding an overall average outdegree of 397.69 assertions per node. Common sense tell us that this figure is beyond dispute. Thanks to our set of extraction tools, we were able to tailor the test data obtained from Advogato to our needs and extract trust networks with specific average outdegree for experimental analysis.

3.4.1. Trust Injection. Trust values $\text{trust}(x)$ computed by the Appleseed metric for source s and node x may differ greatly from explicitly assigned trust weights $W(s, x)$. We have already mentioned before that computed trust ranks may *not* be interpreted as absolute values, but rather in comparison with ranks assigned to all other peers. In order to make assigned rank values more tangible, though, one might expect that tuning the trust injection in^0 to satisfy the following proposition will align computed ranks and explicit trust statements:

$$\forall (s, x) \in E : \ \text{trust}(x)) \in [W(s, x) - \epsilon, W(s, x) + \epsilon]$$

However, when assuming reasonably small ϵ, the approach does not succeed. Recall that *computed* trust values of successor nodes x to s not only depend on assertions made by s, but also on trust ratings asserted by other peers. Hence, perfect alignment of explicit trust ratings with computed ones cannot be accomplished. However, we propose an alignment heuristics, incorporated into Algorithm 4, which proved to work remarkably well in diverse test scenarios. Its basic idea is to add node i and edge (s, i) with $W(s, i) = 1$ to the trust graph $G = (V, E, W)$, treating (s, i) as an indicator to tell whether trust injection in^0 is "good" or not. Latter parameter in^0 has to be adapted in order to make trust(i) converge towards $W(s, i)$. Trust metric computation is hence repeated with different values for in^0 until convergence of explicit and computed trust value for i is achieved. Eventually, edge (s, i) and node i are removed and metric computation is performed one more time. Experiments have shown that our imperfect alignment heuristics yield computed ranks trust(x) for direct successors x of trust source s which come close to previously specified trust statements $W(s, x)$.

```
function Trust_heu ($s \in V$, $d \in [0,1]$, $T_c \in \mathbb{R}^+$) {
  add node $i$, edge $(s,i)$, set $W(s,i) \leftarrow 1$;
  set $\text{in}^0 \leftarrow 20$, $\epsilon \leftarrow 0.1$;
  repeat
    set trust $\leftarrow$ Trust_A ($s$, $\text{in}^0$, $d$, $T_c$);
    $\text{in}^0 \leftarrow$ adapt ($W(s,i)$, trust($i$));
  until trust($i$) $\in [W(s,i) - \epsilon, W(s,i) + \epsilon]$
  remove node $i$, remove edge (s, i);
  return Trust_A ($s$, $\text{in}^0$, $d$, $T_c$);
}
```

Algorithm 4. Adding weight alignment heuristics

3.4.2. Spreading Factor. Small values for d tend to overly reward nodes close to the trust source and penalize remote ones. Recall that low d allows nodes to retain most of the incoming trust quantity for themselves, while large d stresses recommendation of trusted individuals and makes nodes distribute most of the assigned trust to their successor nodes:

Experiment 1 (Impact of parameter d) We compare distributions of computed rank values for three diverse instantiations of d, namely $d_1 = 0.1$, $d_2 = 0.5$, and $d_3 = 0.85$. Our setup is based upon a social network with an average outdegree of six trust assignments and 384 nodes reached by trust energy spreading from our designated trust source. We furthermore suppose $\text{in}^0 = 200$, $T_c = 0.01$, and linear weight normalization. Computed ranks are classified in eleven histogram cells with nonlinear cell

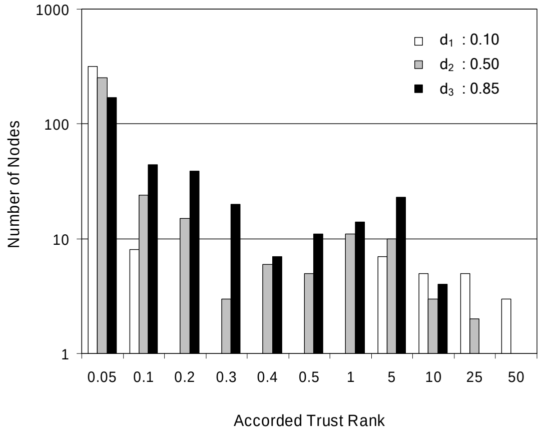

Figure 5. Spreading factor impact

width. Obtained output results are displayed in Figure 5. Mind that we have chosen *logarithmic* scales for the vertical axis in order to render the diagram more legible. For d_1, we may observe that it engenders the highest amount of nodes x with ranks trust$(x) \geq 25$. On the other hand, virtually no ranks ranging from 0.2 to 1 are assigned, while the number of nodes with ranks smaller than 0.05 is again much higher for d_1 than for both d_2 and d_3. Instantiation $d_3 = 0.85$ constitutes the counterpart of d_1. No ranks trust$(x) \geq 25$ are accorded, while interim ranks from between 0.1 and 10 are much more likely for d_3 than for both other instantiations of spreading factor d. Consequently, the number of ranks below 0.05 is lowest for d_3.

The experiment demonstrates that high values for parameter d tend to distribute trust more evenly, neither overly rewarding nodes close to the source, nor penalizing remote ones too rigidly. On the other hand, low d assigns huge trust ranks to very few nodes, namely those which are closest to the source, while the majority of nodes obtains very small trust ranks. We propose to set $d = 0.85$ for general use.

3.4.3. Accuracy and Convergence. We have already mentioned before that the Appleseed algorithm is inherently recursive. Parameter T_c has been introduced as one of two criteria for termination. We will show through an experiment that convergence is reached very fast, no matter how huge the number of nodes trust is flowing through and no matter how large the initial trust injection:

Experiment 2 (Convergence rate) The trust network we consider has an average outdegree of five trust assignments per node. The number of nodes for which

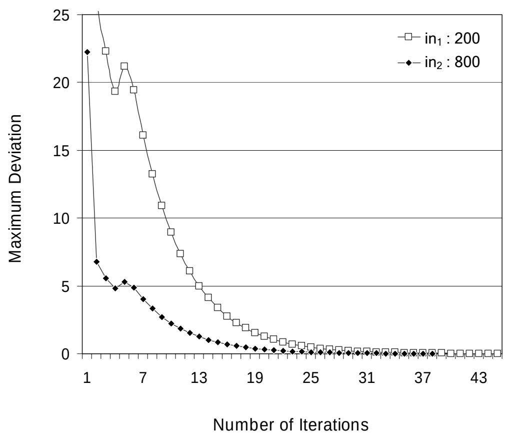

Figure 6. Convergence of Appleseed

trust ranks are assigned amounts to 572. We suppose $d = 0.85$, $T_c = 0.01$, and linear weight normalization. Computation of two runs takes place, one with trust activation $\text{in}_1 = 200$, the other with initial energy $\text{in}_2 = 800$. Figure 6 demonstrates rapid convergence of both runs. Though the trust injection for the second run is four times as high as for the first, convergence is reached in only few more iterations: run one takes 38 iterations, run two terminates after 45 steps.

For both runs, we have assumed accuracy threshold $T_c = 0.01$, which is extremely small and accurate beyond necessity already. However, experience taught us that convergence takes place rapidly even for very large networks and high amounts of trust injected, so that assuming latter value for T_c imposes no scalability issues. In fact, the amount of nodes taken into account for trust rank assignment in the above example well exceeds practical usage scenarios: mind that the case at hand demands 572 documents to be fetched from the Web, complaisantly supposing that these pages containing personal trust information for each node are cached after their first access. Hence, we may well claim that the actual bottleneck of group trust computation is not the Appleseed metric itself, but downloads of trust resources from the network. This bottleneck might also be the reason for selecting thresholds T_c greater than 0.01, in order to make the algorithm terminate faster.

3.5. Implementation and Extensions

Appleseed has been implemented in JAVA, based upon Algorithm 3. We have applied moderate fine-tuning and supplemented our metric with an architectural cushion in order to access "real" machine-readable RDF homepages. Other notable modifications to the core algorithm are discussed briefly:

- **Maximum number of nodes.** We have supplemented the set of input parameters by yet another argument M, which specifies the maximum number of nodes to unfold. This extension hinders trust energy from overly covering vast parts of the entire network. Note that accessing the personal, machine-readable homepages, which contain trust information required for metric computation, represents the actual computation bottleneck. Hence, expanding as few nodes as possible is highly desirable. When choosing reasonably large M, for instance, twice the number of agents assumed trustworthy, we may expect to not miss any relevant nodes: mind that Appleseed proceeds breadth-first and thus considers close nodes first, which are more eligible for trust than distant ones.

- **Upper-bounded path lengths.** Another approach to sensibly restrict the number of nodes unfolded relies upon upper-bounded path lengths. The idea of constraining path lengths for trust computation has been adopted before by Reiter and Stubblebine [31] and within the X.509 protocol [14]. Depending on the overall trust network connectivity, we opt for maximum path lengths between three and six, well aware of Milgram's "six degress of separation" paradigm [27]. In fact, trust decay is inherent to Appleseed, thanks to spreading factor d and backward propagation. Stripping nodes at large distances from the seed therefore only marginally affects trust metric computation results while providing major speed-ups at the same time.

- **Zero trust retention for the source.** Third, we have modified Appleseed to hinder trust source s from accumulating trust energy, essentially introducing one novel spreading factor $d_s = 1.0$ for the seed only. Consequently, all trust is divided among peers of s and none retained, which is reasonable. Remember that s wants to discover trustworthy agents and not assign trust rank to itself. Convergence may increase, since $\text{trust}_{i+1}(x) - \text{trust}_i(x)$ used to be maximal for seed node s, thanks to backward propagation of trust. Furthermore, supposing the same trust quantity in^0 injected, assigned trust ranks become greater in value, also enlarging gaps between neighbors in trust rank.

3.6. Testbed for Local Group Trust Metrics

Trust metrics and models for trust propagation have to be intuitive, underpinning the need for the application of Occam's Razor. Humans must be able to comprehend why agent *a* has been accorded higher trust rank than *b* and come to similar results when asked for personal judgement. Consequently, we have implemented our own testbed, which visually displays social networks, allows zooming of specific nodes and layouts these appropriately, with minimum overlap. Herefore, we relied upon the yFiles [37] library to perform all graph drawing. Moreover, our testbed permits to parameterize Appleseed through dialogs. Detailed output is provided, both graphical and textual. Graphical results comprise highlighting of nodes with trust ranks above certain thresholds, while textual results return quantitative trust ranks of all accessed nodes, numbers of iterations, and so forth. We also implemented the Advogato trust metric and incorporated the latter into our testbed. Hereby, our implementation of Advogato does not require a priori complete trust graph information, but accesses nodes "just in time", similar to Appleseed. All experiments have been conducted on top of the testbed application.

4. Discussion

In this paper, we have introduced various axes to classify trust metrics with respect to various criteria and features. Furthermore, we have advocated the need for local group trust metrics, eventually presenting Appleseed, our main contribution. Through our proposed trust model, we have situated Appleseed within the Semantic Web universe. However, we believe that Appleseed suits other application scenarios likewise, such as, for instance, group trust in online communities and open rating systems. In fact, these system bear several traits similar to the Semantic Web.

Though having discussed numerous design decisions, parameterization proposals and extensions, open issues for future research clearly remain. Further extensions of Appleseed might not only comprise trust and levels of trustworthiness, but also *distrust*, which has been claimed to be different from mere *lack of trust* [23, 28]. Moreover, we have described ranking mechanisms and ways to align direct and indirect, i.e., computed, trust relationships by means of heuristics. Though, an actual policy for eventual *boolean* decisions on which agents to grant trust and which to deny has not been considered. Note that possible criteria are application-dependent. For some, one might want to select the *n* most trustworthy agents. For others, all agents with ranks above given thresholds may be eligible.

At any rate, we strongly believe that local group trust metrics, such as Advogato and Appleseed, will become subject to substantial research for diverse computing domains within the near future.

5. Acknowledgements

We are very grateful to Lule Ahmedi, Nathan Dimmock, Kai Simon and Andrew Twigg for insightful comments and fruitful discussions, which helped us improve the quality of the paper.

References

[1] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *New Security Paradigms Workshop*, pages 48–60, Cumbria, UK, September 1997.

[2] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In H. Paques, L. Liu, and D. Grossman, editors, *Proceedings of the Tenth International Conference on Information and Knowledge Management*, pages 310–317. ACM Press, 2001.

[3] T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open networks. In *Proceedings of the 1994 European Symposium on Research in Computer Security*, pages 3–18, 1994.

[4] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of the 17th Symposium on Security and Privacy*, pages 164–173, Oakland, CA, USA, May 1996. IEEE Computer Society Press.

[5] M. Ceglowski, A. Coburn, and J. Cuadrado. Semantic search of unstructured data using contextual network graphs, June 2003.

[6] R. Chen and W. Yeager. Poblano: A distributed trust model for peer-to-peer networks. Technical report, Sun Microsystems, Santa Clara, CA, USA, February 2003.

[7] E. Dumbill. Finding friends with XML and RDF, June 2002. IBM's XML Watch.

[8] L. Eschenauer, V. Gligor, and J. Baras. On trust establishment in mobile ad-hoc networks. Technical Report MS 2002-10, Institute for Systems Research, University of Maryland, MD, USA, October 2002.

[9] L. Ford and R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, USA, 1962.

[10] Y. Gil and V. Ratnakar. Trusting information sources one citizen at a time. In *Proceedings of the First International Semantic Web Conference*, Sardinia, Italy, June 2002.

[11] J. Golbeck, B. Parsia, and J. Hendler. Trust networks on the semantic web. In *Proceedings of Cooperative Intelligent Agents*, Helsinki, Finland, August 2003.

[12] E. Gray, J.-M. Seigneur, Y. Chen, and C. Jensen. Trust propagation in small worlds. In P. Nixon and S. Terzis, editors, *Proceedings of the First International Conference on Trust Management*, volume 2692 of *LNCS*, pages 239–254. Springer-Verlag, April 2003.

[13] R. Guha. Open rating systems. Technical report, Stanford Knowledge Systems Laboratory, Stanford, CA, USA, 2003.

[14] R. Housely, W. Ford, W. Polk, and D. Solo. Internet X.509 public key infrastructure, January 1999. Internet Engineering Task Force RFC 2459.

[15] A. Jøsang, E. Gray, and M. Kinateder. Analysing topologies of transitive trust. In *Proceedings of the Workshop of Formal Aspects of Security and Trust*, Pisa, Italy, September 2003.

[16] J. Kahan, M.-R. Koivunen, E. Prud' Hommeaux, and R. Swick. Annotea - an open RDF infrastructure for shared web annotations. In *Proceedings of the Tenth International World Wide Web Conference*, pages 623–632, Hong Kong, China, May 2001.

[17] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the Twelfth International World Wide Web Conference*, Budapest, Hungary, May 2003.

[18] M. Kinateder and S. Pearson. A privacy-enhanced peer-to-peer reputation system. In *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies*, volume 2378 of *LNCS*, Prague, Czech Republic, September 2003. Springer-Verlag.

[19] M. Kinateder and K. Rothermel. Architecture and algorithms for a distributed reputation system. In P. Nixon and S. Terzis, editors, *Proceedings of the First International Conference on Trust Management*, volume 2692 of *LNCS*, pages 1–16. Springer-Verlag, April 2003.

[20] R. Levien. *Attack Resistant Trust Metrics*. PhD thesis, UC Berkeley, Berkeley, CA, USA, 2003.

[21] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, Texas, USA, January 1998.

[22] R. Levien and A. Aiken. An attack-resistant, scalable name service, 2000. Draft submission to the Fourth International Conference on Financial Cryptography.

[23] S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, Stirling, UK, 1994.

[24] S. Marsh. Optimism and pessimism in trust. In J. Ramirez, editor, *Proceedings of the Ibero-American Conference on Artificial Intelligence*, Caracas, Venezuela, 1994. McGraw-Hill Publishing.

[25] U. Maurer. Modelling a public key infrastructure. In E. Bertino, editor, *Proceedings of the 1996 European Symposium on Research in Computer Security*, volume 1146 of *Lecture Notes in Computer Science*, pages 325–350. Springer-Verlag, 1996.

[26] H. McKnight and N. Chervany. The meaning of trust. Technical Report MISRC 96-04, Management Informations Systems Research Center, University of Minnesota, MN, USA, 1996.

[27] S. Milgram. The small world problem. In J. Sabini and M. Silver, editors, *The Individual in a Social World - Essays and Experiments*. McGraw Hill, New York, NY, USA, 2nd edition, 92.

[28] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. In *Proceedings of the 35th Hawaii International Conference on System Sciences*, pages 188–196, Big Island, HI, USA, January 2002.

[29] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[30] R. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*, pages 227–270. MIT Press, Boston, CA, USA, 1968.

[31] M. Reiter and S. Stubblebine. Path independence for authentication in large-scale systems. In *ACM Conference on Computer and Communications Security*, pages 57–66, 1996.

[32] M. Reiter and S. Stubblebine. Toward acceptable metrics of authentication. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 10–20, 1997.

[33] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the Second International Semantic Web Conference*, Sanibel Island, FL, USA, September 2003.

[34] K. Sankaralingam, S. Sethumadhavan, and J. Browne. Distributed pagerank for P2P systems. In *Proceedings of the Twelfth International Symposium on High Performance Distributed Computing*, Seattle, Washington, USA, June 2003.

[35] E. Smith, S. Nolen-Hoeksema, B. Fredrickson, and G. Loftus. *Atkinson and Hilgards's Introduction to Psychology*. Thomson Learning, Boston, MA, USA, 2003.

[36] A. Twigg and N. Dimmock. Attack-resistance of computational trust models. In *Proceedings of the Twelfth IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises: Enterprise Security (Special Session on Trust Management)*, pages 275–280, Linz, Austria, June 2003.

[37] R. Wiese, M. Eiglsperger, and M. Kaufmann. yfiles - visualization and automatic layout of graphs. In *Proceedings of the 9th International Symposium on Graph Drawing*, volume 2265 of *LNCS*, pages 453–454, Heidelberg, Germany, January 2001. Springer-Verlag.

[38] P. Zimmermann. *The Official PGP User's Guide*. MIT Press, Boston, MA, USA, 1995.