# Social Network-based Trust in Prioritized Default Logic

**Yarden Katz**
Maryland Information and Network Dynamics Lab
University of Maryland, College Park
8400 Baltimore Ave
College Park, MD 20740
yarden@umd.edu

**Jennifer Golbeck**
Maryland Information and Network Dynamics Lab
University of Maryland, College Park
8400 Baltimore Ave
College Park, MD 20740
golbeck@cs.umd.edu

**Abstract**

A drawback of traditional default logic is that there is no general mechanism for preferring one default rule over another. To remedy this problem, numerous default logics augmented with priority relations have been introduced. In this paper, we show how trust values, derived from web-based social networks, can be used to prioritize defaults. We provide a coupling between the method for computing trust values in social networks and the prioritized Reiter defaults of (Baader & Hollunder 1995), where specificity of terminological concepts is used to prioritize defaults. We compare our approach with specificity-based prioritization, and discuss how the two can be combined. Finally, we show how our approach can be applied to other variants of prioritized default logic.

## Introduction

We are often given conflicting information from distinct sources, forcing us into a decision about what information to accept. This problem is especially complex on the web, where the information sources are many and varied. Our decision in these cases is sometimes reduced to picking the more highly trusted information source. If we think of the information given by sources as a set of default rules, our problem boils down to the following: given defaults from distinct sources which support conflicting conclusions, how should these defaults be prioritized to end up with the most reliable conclusion?

The machinery for expressing priorities between defaults is rich and well-studied, but the question of how these priorities should be generated is frequently left for the user to manually input. When using trust to prioritize defaults, Web-Based Social Networks (WBSNs) offer an accessible source of trust information. We argue that WBSNs can be used to automatically obtain a set of priorities which reflect the user's levels of trust in the information sources.

Within WBSNs, users often reveal information about their relationships with one another. That includes quantitative values representing how much they trust people they know. Using algorithms presented in this work, trust values can be composed to generate recommendations about how much a user should trust an unknown person in the social network. When default rules are asserted on the web and provenance information is available, these trust values can be used to rate the trustworthiness of the source of each default. That can, in turn, be used as a measure of a default's priority.

In this paper, we show how trust values, derived from web-based social networks, can be used to prioritize defaults. We provide a coupling between the method for computing trust values in social networks given in (Golbeck 2005) and the prioritized terminological defaults of (Baader & Hollunder 1995), where specificity of concepts is used to prioritize defaults. We compare our approach with specificity-based prioritization, and discuss how the two can be combined.



## Nonmonotonic Reasoning with Default Rules

When we reason, we often use various rules that are generally but not universally true. For example, we might infer from ($P_1$) The flight is scheduled to leave at 11:00 and ($P_2$) Flights usually leave on time, that we should: ($C$) Be at the airport in time for an 11:00 flight. While it's certainly not true that *every* flight leaves on time, the premise that this is typically true is what licensed our inference. We can formalize a statement such as ($P_2$) using *default rules*. Below we briefly describe Reiter defaults and their simple extension to allow priorities. For the sake of simplicity, we have chosen the account of prioritized defaults given in (Baader & Hollunder 1995). However, our method for combining trust with priorities can be applied to many other variants of defaults.

### Reiter Defaults

A *Reiter default* (henceforth 'default') is of the form:

$$\frac{\alpha \ : \ \beta}{\gamma}$$

where $\alpha, \beta$ and $\gamma$ are formulae of first-order logic. The formula $\alpha$ is the *prerequisite*, $\beta$ the *justification* and $\gamma$ the *consequent*. A default rule can be read intuitively as: *if I can prove the prerequisite from what I believe, and the justification is consistent with what I believe, then add the consequent to my set of beliefs*.

**Definition 1 (Default Theory)** *A default theory $T$ is a pair $\langle \mathcal{W}, \mathcal{D} \rangle$ where $\mathcal{W}$ is a finite set of formulae representing the initial world description (or initial set of beliefs), and $\mathcal{D}$ is a finite set $\{\delta_1, ..., \delta_n\}$ of defaults. $T$ is closed if no free variables appear in either $\mathcal{W}$ or $\mathcal{D}$.*

We will assume for simplicity that free variables in defaults only stand for ground instances. We also, for the sake of exposition, assume that every default has only one justification formula $\beta$, though our approach does not rely on this restriction. On these points, we follow (Baader & Hollunder 1995) where the reader may find the details.

The premise ($P_2$) from our earlier example can be formalized as follows:

$$\delta_f = \frac{Flight(x) \ : \ OnTime(x)}{OnTime(x)}$$

Suppose that $\mathcal{W} = \{Flight(flight714)\}$ and $D = \{\delta_f\}$. Then $\mathcal{W} \vdash Flight(flight714)$, and $\mathcal{W} \cup \{OnTime(flight714)\}$ is consistent, meaning the default $\delta_f$ is *active*. Since $\delta_f$ is active, we apply it and obtain $\mathcal{W} = \mathcal{W} \cup \{OnTime(flight714)\}$. The set $Th(\mathcal{W} \cup \{OnTime(flight714)\})$ is called an *extension*, which we characterize formally below.

**Definition 2 (Reiter Extension)** *Given a set of closed formulae $\mathcal{E}$ and a closed default theory $\langle \mathcal{W}, \mathcal{D} \rangle$, let $E_0 = \mathcal{W}$ and $\forall i \geq 0$ define:*

$$E_{i+1} = \{\gamma \mid \frac{\alpha \ : \ \beta}{\gamma} \in D, \alpha \in Th(E_i) \text{ and } \neg\beta \notin \mathcal{E}\}$$

*Then $\mathcal{E}$ is an R-extension of $\langle \mathcal{W}, \mathcal{D} \rangle$ iff $\mathcal{E} = \bigcup_{i \geq 0} Th(E_i)$*

The above theory has one extension, namely $Th(\mathcal{W} \cup \{Flight(flight714)\})$. Contrast this with the case where $\mathcal{W}$ is:

$$\{Flight(flight714), Delayed(flight714), Delayed(x) \rightarrow \neg OnTime(x)\}$$

In this example, $\mathcal{W} \cup \{OnTime(flight714)\}$ is inconsistent and the inference that $OnTime(flight714)$ is blocked. Thus, this theory has no extension where $OnTime(flight714)$ holds.

## Cases of Conflict

Default rules can conflict. A simple abstract example is when two defaults, $\delta_1$ and $\delta_2$ are applicable (i.e. their justifications are consistent with our knowledge) yet the consequent of $\delta_1$ is inconsistent with the consequent of $\delta_2$. We then typically end up with *two extensions*; one where the consequent of $\delta_1$ holds, and one where the consequent of $\delta_2$ holds. The case of two conflicting defaults is illustrated below, although it is possible to have arbitrarily many conflicting extensions with a larger set of defaults.

**Definition 1 (Chomsky Diamond)** *Let $T = \langle \mathcal{W}, \mathcal{D} \rangle$ and $\mathcal{W} = \{Professor(chomsky), Activist(chomsky)\}$, $\mathcal{D} = \{\delta_1, \delta_2\}$, where:*

$$\delta_1 = \frac{Professor(x) \ : \ Passive(x)}{Passive(x)}$$

$$\delta_2 = \frac{Activist(x) \ : \ \neg Passive(x)}{\neg Passive(x)}$$

Note that $T$ has two extensions, $\mathcal{E}_1$ and $\mathcal{E}_2$. In one,

$$\neg Passive(chomsky) \in \mathcal{E}_1$$

while in the other,

$$Passive(chomsky) \in \mathcal{E}_2.$$

It is often desirable to resolve conflicting defaults like $\delta_1$ and $\delta_2$. This can be done by introducing *priorities*. Given a priority relation $>$, we interpret $\delta_2 > \delta_1$ to mean that $\delta_2$ has higher priority than $\delta_1$.

**Definition 3 (Prioritized Default Theory)** *A prioritized default theory $\mathcal{T}$ is a triple $\langle \mathcal{W}, \mathcal{D}, < \rangle$, where $\mathcal{W}, \mathcal{D}$ are as usual, and $<$ is a partial ordering on $\mathcal{D}$.*

A prioritized version of $T$ would be $\mathcal{T} = \langle \mathcal{W}, \mathcal{D}, < \rangle$. It is easy to see that if $\delta_2 > \delta_1$, then $\mathcal{E}_2$ should not be an extension of $\mathcal{T}$. The reason is that since $\delta_2$ has higher priority, it should be applied first, which in turns blocks the application of $\delta_1$. The definition formalizing this intuition, following (Baader & Hollunder 1995) again, is given below.

**Definition 4** *Let $\mathcal{T} = \langle \mathcal{W}, \mathcal{D}, < \rangle$ be a prioritized default theory, and $\mathcal{E}$ a set of formulae. Let $\mathcal{E}_0 = \mathcal{W}$, and $\forall i \geq 0$ define:*

$$E_{i+1} = E_i \ \cup \ \{\gamma \mid d = \frac{\alpha \ : \ \beta}{\gamma} \in \mathcal{D}, \alpha \in Th(E_i), \neg\beta \notin \mathcal{E}, \text{ and every } d' > d \text{ is not active in } E_i\}$$

*Then $\mathcal{E}$ is a P-extension of $\langle \mathcal{W}, \mathcal{D}, < \rangle$ iff $\mathcal{E} = \bigcup_{i \geq 0} Th(E_i)$*

It is easy to see now that in the above example, if $\delta_2 > \delta_1$, then $\mathcal{E}_2$ is not an extension. Similarly, if $\delta_1 > \delta_2$ were true, then $\mathcal{E}_1$ would not be an extension.

There have been many other approaches to prioritized default logic, where a priority relation is introduced in either the object or the meta language. We refer the reader to (Delgrande & Schaub 2000) for an extensive survey.

Regardless of the specifics of a given approach, some kinds of priority relations are undesirable. In particular, it is unrealistic to require the priority relation to be a total ordering over the defaults, especially if we are dealing with a large and changing collection of defaults. We follow the more common and flexible approach which only requires the priority relation to be a partial ordering.

In previous approaches, the priority relation was usually taken as a given, and sometimes compiled into the object language and reasoned over. In contrast, our priorities are based on the trust rating of the sources of the defaults−i.e. their creators−in a web-based social network. The next section introduces the concept of trust in web-based social networks, and a corresponding algorithm for computing trust ratings. In section  we apply this work to the case of prioritizing defaults.

## Trust in Web-based Social Networks

Web-based social networks (WBSNs) are online communities where users maintain lists of people they know. Other users can browse those connections, and access contact and profile information about people in the network. The popularity of WBSNs has grown dramatically over the last few years, with hundreds of networks that have hundreds of millions of members. Within WBSNs, a variety of features are available to allow users to annotate their relationship; trust is one of these.

When trust is assigned on a quantitative scale, we can make computations with trust values in the network. If we choose a specific user and look at all of the trust ratings assigned to that person, we can see the average opinion about the person's trustworthiness. Trust, however, is a subjective concept where averages are often unhelpful. Consider the simple example of asking weather the President is trustworthy. Some people believe very strongly that he is, and others believe very strongly that he is not. In this case, the average trust rating is not useful to either group. However, given provenance information about the trust annotations, we can significantly improve on the average case. If someone (the $source$) wants to know how much to trust another person (the $sink$), we can look at the who trusts the sink, see how much the source trusts the intermediate people, and produce a result that weights ratings from trusted people more highly than those from untrusted people.

In this section, we present a description of and algorithm for inferring trust values, and show how the results can be applied.

### Background and Related Work

We present an algorithm for inferring trust relationships in social networks, but this problem has been approached in several ways before. Here, we highlight some of the major contributions from the literature and compare and contrast them with our approach.

Trust has been studied extensively in peer-to-peer systems including (Kamvar, Schlosser, & Garcia-Molina 2004), (Aberer & Despotovic 2001), (Lee, Sherwood, & Bhattacharjee 2003). There are basic differences in the meaning of trust in P2P networks and social networks that makes these algorithms inappropriate for social use. In P2P systems, trust is a measure of performance, and one would not expect the performance of $peer_a$ to be very different when it is interacting with $peer_b$ vs. $peer_c$. Thus, one global recommendation about the trustworthiness of $peer_a$ will usually be sufficient. Socially, though, two individuals can have dramatically different opinions about the trustworthiness of the same person. Our algorithms intentionally avoid using a global trust value for each individual to preserve the personal aspects that are foundations of social trust.

There are several algorithms for computing trust in social networks specifically. A thorough treatment can be found in (Golbeck 2005). Our algorithm differs from most existing algorithms in one of three major ways: we output recommendations in the same scale that users assign trust (vs. eigenvector based approaches like (Ziegler & Lausen 2004)), our computations are about people (vs. trust in statements as in (Richardson, Agrawal, & Domingos 2003)), and we create personalized recommendations (vs. global ratings as are used in P2P systems and (Levin & Aiken 1998)).

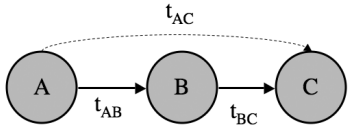


Figure 1: An illustration of direct trust values between nodes A and B ($t_{AB}$), and between nodes B and C ($t_{BC}$). Using a trust inference algorithm, it is possible to compute a value to recommend how much A may trust C ($t_{AC}$).

### TidalTrust: An Algorithm for Inferring Trust

When two individuals know each other, they can assess the trustworthiness of one another. Two people who are not directly connected do not have a foundation for knowing about trust. However, the paths connecting them in the network contain information that can be used to infer how much they may trust one another.

For example, consider that Alice trusts Bob, and Bob trust Charlie. Although Alice does not know Charlie, she knows and trusts Bob who, in turn, has information about how trustworthy he believes Charlie is. Alice can use information from Bob and her own knowledge about Bob's trustworthiness to infer how much she may trust Charlie. This is illustrated in Figure 1.

Our algorithm looks at the trust values along paths connecting the source and sink to compute a recommendation to the source about how much to trust the sink. When making this computation, several features of the network and paths must be considered to produce the most accurate results. In this section, we describe how path length and trust values on paths affect the computations, and how these features were incorporated into our algorithm.

**Incorporating Path Length** A limit on the depth of the search should lead to more accurate results, since previous work (Golbeck 2005)has shown that average error (measured as the absolute difference between the computed rating and the user's rating - call this $\overline{\Delta}$) increases as depth increases. This intuitively makes sense: getting information from one intermediate person should usually be more reliable than information passed down a long chain of people. Accuracy decreases as path length increases, and thus shorter paths are more desirable. However, the tradeoff is that fewer nodes will be reachable if a fixed limit is imposed on the path depth. To balance these factors, we use the shortest search depth that will produce a result. This preserves the benefits of a shorter path length without limiting the number of inferences that can be made.

**Incorporating Trust Values** Previous research(Ziegler & Golbeck 2006) also indicates that the most accurate information will come from the most highly trusted neighbors.

| Algorithm | | |
|---|---|---|
| Network | TidalTrust | Simple Average |
| Trust Project | 1.09 | 1.43 |
| FilmTrust | 1.35 | 1.93 |

Thus, we set a minimum trust threshhold and require only consider paths where all edges have trust ratings at or above the threshhold. We want to include only the highest trust ratings possible (ignoring paths that have low values) without limiting the number of inferences that can be made (because the threshhold may be so high that no paths exist). We define a variable $max$ that represents the largest trust value that can be used as a minimum threshold such that a path can be found from source to sink.

**Full Algorithm for Inferring Trust** Incorporating the elements presented in the previous sections, the final TidalTrust algorithm can be assembled. The name was chosen because calculations sweep forward from source to sink in the network, and then pull back from the sink to return the final value to the source.

$$t_{is} = \frac{\sum_{j \in adj(j) \mid t_{ij} \geq max} t_{ij} t_{js}}{\sum_{j \in adj(j) \mid t_{ij} \geq max} t_{ij}} \quad (1)$$

TidalTrust is a modified breadth-first search. The source's inferred trust rating for the sink ($t_{source,sink}$) is a weighted average if the source's neighbors' ratings of the sink (see Forumula 1). The source node begins a search for the sink. It will poll each of its neighbors to obtain their rating of the sink. If the neighbor has a direct rating of the sink, that value is returned. If the neighbor does not have a direct rating for the sink, it queries all of its neighbors for their ratings, computes the weighted average as shown in Formula 1, and returns the result. Each neighbor repeats this process, keeping track of the current depth from the source. Each node will also keep track of the strength of the path to it, computed as the minimum of the source's rating of the node and the node's rating of its neighbor. The neighbor records the maximum strength path leading to it. Once a path is found from the source to the sink, the depth is set at the maximum depth allowable. Since the search is proceeding in a Breadth First Search fashion, the first path found will be at the minimum depth. The search will continue to find any other paths at the minimum depth. Once this search is complete, the trust threshold ($max$) is established by taking the maximum of the trust paths leading to the sink. With the $max$ value established, each node completes the calculations of a weighted average by taking information from nodes that they have rated at or above the $max$ threshold. Those values are passed back to the neighbors who queried for them, until the final result is computed at the source.

### Accuracy of TidalTrust

As presented above, TidalTrust strictly adheres to the observed characteristics of trust: shorter paths and higher trust values lead to better accuracy. However, there are some things that should be kept in mind. The most important is that networks are different. Depending on the subject (or lack thereof) about which trust is being expressed, the user community, and the design of the network, the effect of these properties of trust can vary. While we should still expect the general principles to be the same−shorter paths will be better than longer ones, and higher trusted people will agree with us more than less trusted people−the proportions of those relationships may differ from what was observed in the sample networks used in this research.

There are several algorithms that output trust inferences, but none of them produce values within the same scale that users assign ratings. Some trust algorithms form the Public Key Infrastructure (PKI), such as Beth-Borcherding-Klein (Beth, Borcherding, & Klein 1994), are more appropriate for comparison. Due to space limitations that comparison is not included here, but can be found in (Golbeck 2005). One direct comparison to make is to compare the $\overline{\Delta}$ from TidalTrust to the $\overline{\Delta}$ from taking the simple average of all ratings assigned to the sink as the recommendation. We made this comparison using two real world networks. As shown in table, the TidalTrust recommendations outperform the simple average in both networks, and these results are statistically significant with $p < 0.01$.

## Basing Priority on Trust Values

Given a social network, an ordinary default theory $T$, and a source node $Src$ in the network, we can now now prioritize the defaults according to trust values.

### Algorithm

**procedure** $TrustPrioritize(W, D, Src, Prov)$:
**Input:**
(1) A set of initial formulae $W$
(2) A source node $Src$
(3) A set $D = \{\delta_1, ..., \delta_n\}$ of defaults,
(4) A function $Prov : D \rightarrow Nodes$
**Return:**
A set of extensions
$P := \emptyset$
**for** every $d, d' \in D$:
**if** $TidalTrust(Src, Prov(d)) < TidalTrust(Src, Prov(d'))$:
$P = P \cup \{d < d'\}$
**if** $Prov(d) = Src$ **and** $Prov(d') \neq Src$:
$P = P \cup \{d' < d\}$
**return** $ComputeExtensions_{\mathcal{PL}}(W, D, P)$

The simple algorithm for generating extensions based on trust values is given below. Note that our method does not make any assumptions about the specifics of the base default logic language $\mathcal{PL}$. We do, however, assume the following are available:

1. A function $ComputeExtensions_{\mathcal{PL}}$ for computing the extensions of $\mathcal{PL}$, which takes a prioritized default theory as input.
2. A *source node*, which in our case is the node according to which priorities will be generated. Intuitively, this can be

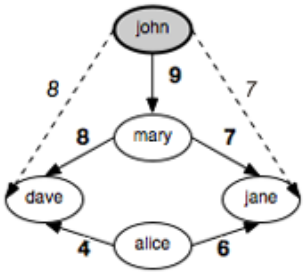


Figure 2: The social network between John, Mary, Dave, Jane and Alice

thought of as our 'viewpoint' in the social network−we reason from the perspective of the source node.

If restricted to normal form, any prioritized default theory of (Baader & Hollunder 1995) is always guaranteed to have an extension. In addition, every prioritized normal default extension is also a Reiter extension. Since we have not in any way changed the semantics of the prioritized defaults, it is obvious that the same desirable properties hold true for our approach. For this reason, we restrict ourselves to normal defaults for the remainder of the paper.

### Example: Using Trust for Choosing a Film

Suppose that we are dealing with a film knowledge base. A group of friends−John, Mary, Dave, Jane and Alice−each input their film preferences, such as preferred genre or directors/actors, in the form of default rules. Their preferences are as follows:

$$\mathcal{W} = \{IndieFilm(hce), SpanishFilm(hce), \\ DirectedBy(hce, Almodovar)\}$$
$$\mathcal{D} = \{\delta_{john}, \delta_{dave}, \delta_{jane}\}$$
$$\delta_{john} = \frac{Comedy(x)}{\neg Watch(x)}$$
$$\delta_{jane} = \frac{IndieFilm(x) \wedge SpanishFilm(x)}{\neg Watch(x)}$$
$$\delta_{dave} = \frac{IndieFilm(x) \wedge Directed(x, Almodovar)}{Watch(x)}$$

We assume that every Spanish film is a film, and similarly that every film directed by anyone (in our case, Almodovar) is also a film.

In our scenario, John, Mary, Dave and Alice are part of a social network, shown in Figure 2. The direct trust values between two nodes in the network are given in bold, while inferred trust values are italicized and are shown as a dotted edge.

Suppose that John is trying to decide whether or not he should watch the film $hce$, the only film currently in our knowledge base. John's only preference is not to watch comedies, which does not apply to $hce$. Simply looking at the defaults in $\mathcal{D}$, a conflict arises. According to $\delta_{jane}$, John should not watch the movie since it is a Spanish film. On the other hand, according to $\delta_{dave}$, John should watch the film since it is directed by Almodovar.

Note that John did not directly rate Dave and Jane. John's only connection to the two is via Mary, who he highly trusts. Mary does not have any film preferences, and so we cannot use her to resolve the conflict. According to TidalTrust, John's inferred trust values for Dave and Jane are 8 and 7, respectively. Thus, the relevant priority yielded in this case is $\delta_{jane} < \delta_{dave}$, which allows John to conclude that he should watch $hce$.

Consider the same scenario, except this time with Alice as the source node. Unlike John, Alice has direct trust ratings for Dave and Jane, and unlike Mary, Alice has stated that Jane is more trusted than Dave Therefore, there will be an extension where Alice's conclusion, based on the generated priorities $\delta_{dave} < \delta_{jane}$, is *not* to watch $hce$. Clearly, this extension is not possible if we pick John as the source node, differentiating between the two nodes' relations to the rest of the social network.

## Discussion and Conclusions

### Priority of the Source Node

Cases can arise where the source node has a default that conflicts with another node's default. In our approach, we chose to prioritize the defaults of the source higher than the defaults of other nodes in the social network. This is reflected in the algorithm, where we explictly add to the default theory that the defaults associated with the source have higher priority than all others. We believe this is the most appropriate choice for the case when dealing with social networks.

If the choice to explicitly prefer the source's defaults is not made, then new cases of conflict can arise. Consider the following abstract example. Suppose we have a root node $A$ with an edge $AB$. Assume that $A$ has one default whose consequent is $\varphi(x)$, i.e. $\delta_A = \frac{\top}{\varphi(x)}$, and that $B$ has one default $\delta_B = \frac{\top}{\neg\varphi(x)}$. Regardless of the value $t_{AB}$ (or the value of any other edges $A$ might have) we are guaranteed to have an extension where $\varphi(x)$ holds. The reason is that $A$ does not necessarily have an explicit trust rating for itself, i.e. there is no $t_{AA}$ value. Note that this is very different from the usual reason for why $\delta_A$ and $\delta_B$ would generate two extensions in ordinary default logic. Therefore, in systems where this value is not present or assumed, it seems there is no way to determine the priority of $\delta_A$ compared with other defaults in the system. This issue will arise whenever the source node has an applicable default whose consequent might conflict with defaults of other nodes in the system.

In such cases, at least two simple resolutions are possible:

1. Make the assumption that the source node has "infinite" credibility−i.e. one always trusts oneself over all others, or alternatively,
2. Make the assumption that when getting a recommendation from other nodes, one should ignore one's own preferences.

In our approach, the first choice was made. We contrast this with the case where specificity is used as a measure of priority.

### Priority and Specificity

In (Baader & Hollunder 1995), priorities between defaults are induced by the specificity of their justifications. While this approach is useful, it cannot resolve every case. In our first example where John is the source node, a specificity-based approach will not decide between Dave's default rule and Alice's. In this case, our approach can be used to *supplement* the priorities generated by specificity-based approach.

Going back to the issue raised by the preferences of the source in the film example, we see that specificity might be altogether inappropriate. For example, suppose that John is the source node and we know that in general his preference not to watch any film that is a comedy. Let's assume that we have one given film, $c$, and that $RomanticComedy(c)$ and $RomanticComedy(x) \rightarrow Comedy(x)$. In this case, it does not make sense for John's choice to not watch $c$, based on his preference, to be defeated by another node $X$, where $\delta_X = \frac{RomanticComedy(x)}{Watch(x)}$ simply because $\delta_X$ is more specific. John's preference, while defined more generally than that of node $X$, should still apply.

In the Tweety triangle, specificity clearly leads to the desirable extension. In fact, whenever dealing with a set of defaults that are meant to *classify* objects and their properties most accurately, the specificity-based approach is generally more appropriate. However, as we have shown, such an approach may fail if we use a set of defaults to express user preference.

In summary, we have presented a preliminary coupling between traditional default logic with priorities and a method for inferring trust in web-based social networks. We argue that the latter provides a good way to generate priorities for default rules. This approach makes it possible to make use of the many large and readily available existing web-based social networks, thus grounding the priorities in real web data. Such an approach differs from the more traditional approaches to priority, where the priorites are taken as specifically tailored to the set of defaults at hand.

While the more traditional approach is appropriate for closed knowledge representation systems, our approach reuses existing web data, which makes the introduction of prioritized defaults into established web systems less demanding. Furthermore, we emphasize that in a system where default rules use a different mechanism for priorities, user preferences, encoded as a web-based social network, can be used as an alternative. That is, when the first mechanism of priority might be incomplete, the priorities generated from the social network can be used to possibly fill the gap. In addition, we have also highlighted a case where a specificity-based approach is likely to be inappropriate, and where a trust value based approach shows more promise.

## Future Work

The quality of the results obtained by prioritizing with trust can be determined empirically when they are applied within applications. One of the main networks we have used for testing is part of the FilmTrust system. FilmTrust currently uses inferred trust values to compute predictive movie ratings customized to each user based on who they trust. However, the current system does not allow for users to specify any default rules about their preferences. Such a default rule system fits well in the context of films.

As part of our future work, we will be deploying a rule system in the FilmTrust system[1], a social network about movies. These defaults will be used in two ways. First, they can help tailor recommendations for the user who asserted rules. They can also be used to filter recommendations for others who trust the user who asserted the rules. In this application, it will be common for defaults to conflict. In such cases, trust is an obvious option for determining which rules to apply.

This will allow us to quantitatively and qualitatively measure the performance of using trust for prioritizing defaults. Showing that the trust-prioritized defaults improve performance will validate how our approach can be used to develop intelligent applications.

## References


Aberer, K., and Despotovic, Z. 2001. Managing trust in a peer2peer information system. *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM01)* 301–317.

Baader, F., and Hollunder, B. 1995. Priorities on Defaults with Prerequisites, and Their Application in Treating Specificity in Terminological Default Logic. *J. Autom. Reasoning* 15(1):41–68.

Beth, T.; Borcherding, M.; and Klein, B. 1994. Valuation of trust in open networks. *Proceedings of ESORICS 94*.

Delgrande, J. P., and Schaub, T. 2000. Expressing preferences in default logic. *Artif. Intell.* 123(1-2):41–87.

Golbeck, J. 2005. *Computing and Applying Trust in Web-based Social Networks*. Ph.D. Dissertation, University of Maryland, College Park.

Kamvar, S. D.; Schlosser, M. T.; and Garcia-Molina, H. 2004. The eigentrust algorithm for reputation management in p2p networks. *Proceedings of the 12th International World Wide Web Conference*.

Lee, S.; Sherwood, R.; and Bhattacharjee, B. 2003. Cooperative peer groups in nice. *IEEE Infocom*.

Levin, R., and Aiken, A. 1998. Attack resistant trust metrics for public key certification. *7th USENIX Security Symposium*.

Richardson, M.; Agrawal, R.; and Domingos, P. 2003. Trust management for the semantic web. *Proceedings of the Second International Semantic Web Conference*.

Ziegler, C.-N., and Golbeck, J. 2006. Investigating Correlations of Trust and Interest Similarity. *Decision Support Services*.

Ziegler, C., and Lausen, G. 2004. Spreading activation models for trust propagation. *Proceedings of the IEEE International Conference on e-Technology, e-Commerce, and e-Service*.


[1] http://trust.mindswap.org/FilmTrust/