System Architecture for Trust-Based News Recommenders on the Web

Cristian E. Briguez[1,2], Fernando M. Sagui[1,2],
Marcela Capobianco[1,2], and Ana G. Maguitman[1,2]

[1] Laboratorio de Investigación y Desarrollo en Inteligencia Artificial
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur, Av. Alem 1253, (B8000CPB) Bahía Blanca, Argentina
{ceb, fms, mc, agm}@cs.uns.edu.ar
[2] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)

Abstract. A fundamental challenge in personalized news recommendation is to account for the notion of trust. In this work we show how the notion of trust can be qualitatively modeled and incorporated into the architecture of a news recommendation system. The proposed system enables users to express explicit trust statements on news reports, news sources and other users. Trust is modeled and propagated using a dialectical process supported by a defeasible logic programming interpreter. We illustrate the operation of the system with some illustrative examples, describe the system architecture and discuss future steps.

Keywords: argumentation, news recommender systems, trust management

1 Introduction

We frequently seek suggestions from people we trust for deciding the best place to acquire some service or the best source to obtain information about certain topic. Suggestions of people we trust may also help decide who else to trust. The Web offers new opportunities to create recommendation services based on trust. In particular, news management systems on the Web can take advantage of the large community of readers to rank news, determine the reputation of an information source or propagate trust among users. This can help decide which news are more interesting or trustworthy to certain user, providing more personalized services.

The dynamics of news credibility has mainly been studied through quantitative approaches (e.g. [8]). However, a purely quantitative perspective to news credibility has several limitations. In particular, quantitative approaches make it hard to provide readers with a justification of why certain news should be trusted, or might been unable to deal with subtle notions such as distrust or trust repair.

A more appealing approach would consist in combining quantitative and qualitative criteria to filter and rank news. In this sense, quantitative methods can help determine if the news topic is relevant to the user's interest, while

qualitative criteria could help decide whether the news comes from a reliable source. For example, a qualitative approach is more natural to deal with some properties of trust such as being context dependent, subjective, asymmetrical, dynamic and not always transitive.

This paper extends previous work on recommendation technologies presented in [3] by describing the architecture of a Trust-Based Recommendation system for web news and its implemented components. A key ingredient of the proposed architecture is the propagation of trust based on inference mechanisms. In particular, a defeasible logic programming interpreter is used to manage interpersonal trust and distrust. We demonstrate the use of the tool with a set of illustrative examples.

2 Background

2.1 Recommendation systems and trust

Recommendation systems are support mechanisms that assist users in their decision-making process while interacting with large or complex information spaces. They attempt to generate a model of the user or user's task and apply diverse heuristics to anticipate what information may be of interest to the user. In order to come up with recommendations, conventional recommender systems rely on similarity measures between users or contents, computed on the basis of methods coming either from the information retrieval or the machine learning communities. Recommender systems adopt mainly two different views to help predict information needs. The first approach is known as *user modeling* and relies on the use of a profile or model of the users, which can be created by observing users' behavior (e.g., [7]). The second approach is based on *task modeling*, where recommendations are based on the context in which the user is immersed (e.g., [1]). The context may consist of an electronic document the user is editing, web pages the user has recently visited, etc.

Two main techniques have been used to compute recommendations: *content-based* and *collaborative filtering*. Content-based recommenders [10] are driven by the premise that user's preferences tend to persist through time. These recommenders frequently use machine-learning techniques to generate a profile of the active user, typically stored as a list of rated items. In order to determine if a new item is a potentially good recommendation, content-based recommender systems rely on similarity measures between the new items and the rated items stored as part of the user model. On the other hand, recommender systems based on collaborative filtering [12] are based on the assumption that users' preferences are correlated. These systems maintain a pool of users' profiles associated with items that the users rated in the past. For a given active user, collaborative recommender systems find other similar users whose ratings strongly correlate with the current user. New items not rated by the active user can be presented as suggestions if similar users have rated them highly.

Trust is a fundamental concept in human behavior, which for many years has enabled collaboration. Therefore, trust is an important aspect in the implemen-

tation of recommendation systems. Typically, the notion of trust is defined in terms of two components: trusting intentions and trusting beliefs. For example, a user can trust the intentions of a vendor or the intentions of a service or information provider. On the other hand, a user can trust the beliefs of other users. Trust models have been applied in a number of areas, such as E-commerce [9], Social Networks [13] and P2P systems [4]. Many proposals have addressed the notion of trust from a formal perspective [2] while others take a more empirical approach [6].

2.2 Defeasible logic programming

Defeasible logic programming (DeLP) [5] is a general-purpose defeasible argumentation formalism based on logic programming, intended to model inconsistent and potentially contradictory knowledge. This formalism provides a knowledge representation language which gives the possibility of representing tentative information in a declarative manner, and a reasoning mechanism with considers all ways a conclusion could be supported and decides which one has the best support. A defeasible logic program has the form $\mathcal{P} = (\Pi, \Delta)$, where Π and Δ stand for *strict* and *defeasible* knowledge, respectively. The set Π involves *strict rules* of the form $P \leftarrow Q_1, \ldots, Q_k$ and *facts* (strict rules with empty body), and it is assumed to be *noncontradictory* (i.e., no complementary literals P and $\sim P$ can be inferred, where $\sim P$ denotes the contrary of P). The set Δ involves *defeasible rules* of the form $P \prec Q_1, \ldots, Q_k$, which stand for *"$Q_1, \ldots Q_k$ provide a tentative reason to believe P."* Rules in DeLP are defined in terms of *literals*. A literal is an atom A or the strict negation ($\sim A$) of an atom. Default negation (denoted not A) is also allowed in the body of defeasible rules (see [5] for details).

Deriving literals in DeLP results in the construction of *arguments*. An argument $\mathcal{A}$ for a literal Q (denoted $\langle \mathcal{A}, Q \rangle$) is a (possibly empty) set of ground defeasible rules that together with the set Δ provide a proof for a given literal Q, satisfying the additional constraints of *noncontradiction* (*i.e.*, an argument should not allow the derivation of contradictory literals) and *minimality* (*i.e.*, the set of defeasible information used to derive Q should be minimal). Note that arguments are obtained by a mechanism similar to the usual query-driven SLD derivation from logic programming, performed by backward chaining on *both* strict and defeasible rules; in this context a negated literal $\sim P$ is treated just as a new predicate name *no* P. In DeLP, arguments provide tentative support for claims (literals). Clearly, as a program $\mathcal{P}$ represents incomplete and tentative information, an argument $\langle \mathcal{A}, Q \rangle$ may be *attacked* by other arguments also derivable from $\mathcal{P}$. An argument $\langle \mathcal{B}, R \rangle$ is a *counter-argument* for $\langle \mathcal{A}, Q \rangle$ whenever a subargument $\langle \mathcal{A}', Q' \rangle$ (with $\mathcal{A}' \subseteq \mathcal{A}$) in $\langle \mathcal{A}, Q \rangle$ can be identified, such that $\langle \mathcal{B}, R \rangle$ and $\langle \mathcal{A}', Q' \rangle$ cannot be simultaneously accepted since their joint acceptance would allow contradictory conclusions to be inferred from $\Pi \cup \mathcal{A}' \cup \mathcal{B}$. If the attacking argument $\langle \mathcal{B}, R \rangle$ is preferred over $\langle \mathcal{A}', Q' \rangle$, then $\langle \mathcal{B}, R \rangle$ is called a *defeater* for $\langle \mathcal{A}, Q \rangle$. The preference criterion commonly used is *specificity* [5], preferring those arguments which are more direct or more informed, although other criteria could be adopted.

In DeLP the search for defeaters for a given argument $\langle \mathcal{A},Q \rangle$ prompts a recursive process, resulting in the generation of a *dialectical tree*: the root node of this tree is the original argument at issue, and every children node in the tree is a defeater for its parent. Additional restrictions help to avoid circular situations when computing branches in a dialectical tree, guaranteeing that every dialectical tree is finite (see [5] for details). Nodes in the tree can be marked either as *defeated* (D-nodes) or as *undefeated* (U-nodes). The marking of the dialectical tree is performed as in an AND-OR trees: leaves are always marked as undefeated nodes (as they have no defeaters); inner nodes can be be marked either as undefeated (if and only if every of its children nodes is marked as defeated) or as defeated (whenever at least one of its children has been marked as undefeated). The original argument $\langle \mathcal{A},Q \rangle$ (the root of tree) is deemed as ultimately acceptable or *warranted* whenever it turns out to be marked as undefeated after applying the above process.

Note also that the computation of the dialectical tree is performed automatically by the DeLP interpreter on the basis of the program available. This process is based on an abstract machine which extends Warren's abstract machine for PROLOG[5]. Given a DeLP program $\mathcal{P}$, solving a query Q with respect to $\mathcal{P}$ may result in four possible answers: YES (there is at least one warranted argument $\mathcal{A}$ for Q); NO (there is at least one warranted argument $\mathcal{A}$ for $\sim Q$); UNDECIDED (none of the previous cases hold); and UNKNOWN (Q is not present in the program signature). The emerging semantics is skeptical, computed by DeLP on the basis of the goal-directed construction and marking of dialectical trees, which is performed in a depthfirst fashion. Additional facilities (such as visualization of dialectical trees, zoom-in/zoom-out view of arguments, etc.) are integrated in the DeLP environment to facilitate user interaction when solving queries.

3 The proposed news recommendation system

In this paper we present a practical implementation of a trust model for news recommendation and analyze how it behaves in real life applications. Our proposal takes as a starting point a set of postulates for trust previously reported in [11] and shows how to incorporate them into the architecture of a recommender. Simply put, our system deals with three different entities: *viewers*, *reports* and *sources*.

A *news article* or *report* is a written communication of a news event prepared by a specific news agency (source). When a report is made available on the Web, we can identify fields such as *title*, *source*, *timestamp*, *description*, *category* and *link to news content*. Other information related to the report such as *author* can also be derived in certain situations.

The *source* of a news article is the agency in charge of supplying the report to be used by the media. News can also be published by social networks, web pages or blogs.

A *viewer* is a user of the news service. The system maintains a pool of viewers. Viewers can also provide trust statements about reports, sources and other viewers.

We have identified a fundamental relation among these entities, needed to model the concept of trust, which we have called *Trust/Distrust Statements*. A trust (distrust) statement is an explicit assertion of the fact that a viewer trusts (distrusts) a report, a source or another viewer. These statements allow to infer implicit trust relations, which are useful to provide recommendations to the viewer based on trust.

3.1 Using DeLP to model news trust

We will use the following set of postulates (previously developed in [11]) to model the notion of trust among users, news reports and news sources in an intuitive way.

Postulate 1. A report coming from a trusted source will typically be trusted.
Postulate 2. A report coming from a distrusted source will typically be distrusted.
Postulate 3. A report trusted by a trusted viewer will typically be trusted.
Postulate 4. A report distrusted by a trusted viewer will typically be distrusted.
Postulate 5. A source trusted by a trusted viewer will typically be trusted.
Postulate 6. A source distrusted by a trusted viewer will typically be distrusted.
Postulate 7. A report coming from a trusted source will typically be trusted, even if it is distrusted by a trusted viewer.
Postulate 8. A report coming from a distrusted source will typically be distrusted, even if it is trusted by a trusted viewer.

These postulates can be translated into the following DeLP rules:

$$
\begin{aligned}
trust\_report(V,R) &\prec report\_source(R,S), trust\_source(V,S)\\
\sim trust\_report(V,R) &\prec report\_source(R,S), \sim trust\_source(V,S)\\
trust\_report(V,R) &\prec trust\_viewer(V,V_1), trust\_report(V_1,R)\\
\sim trust\_report(V,R) &\prec trust\_viewer(V,V_1), \sim trust\_report(V_1,R)\\
trust\_source(V,S) &\prec trust\_viewer(V,V_1), trust\_source(V_1,S)\\
\sim trust\_source(V,S) &\prec trust\_viewer(V,V_1), \sim trust\_source(V_1,S)\\
trust\_report(V,R) &\prec report\_source(R,S), trust\_source(V,S),\\
&\quad\ trust\_viewer(V,V_1), \sim trust\_report(V_1,R)\\
\sim trust\_report(V,R) &\prec report\_source(R,S), \sim trust\_source(V,S),\\
&\quad\ trust\_viewer(V,V_1), trust\_report(V_1,R)
\end{aligned}
$$

3.2 System architecture

Based on the model presented before, we have implemented a novel system for news recommendations, based on a particular architecture that we describe next. We have chosen to use a client-server style architecture, given that it can accommodate our requirements in a natural way. Figure 1 depicts the main components of the system.

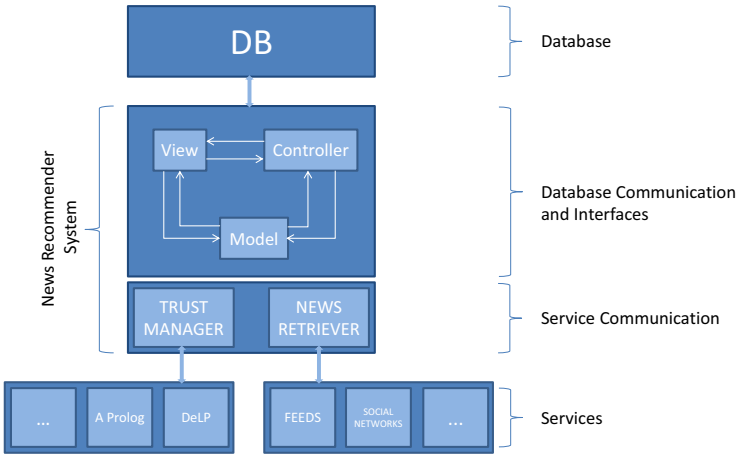

Fig. 1. Proposed architectural pattern for a news recommendation system.

At this point, we work with two different types of services. The first one provides news that were obtained from the available sources. The second one deals with news trust. This service must be able to decide if a certain news could be trustworthy for the current logged in user. So far, we have focused on the implementation of the second service, using a DeLP interpreter. Note that even tough the trust manager is currently implemented using DeLP, the system is designed to allow a seamless transition to a different system, given that it was built taking into account modularity as a fundamental design principle.

The recommender system was implemented as a web application, because web applications can be easily accessed from any computer or location equipped with a browser and Internet access (accessibility and portability). Most of the work is done on the server, resulting in low resource consumption and giving the application independence from operating system to avoid compatibility problems (efficiency and multi-platform). Also the ability to update and maintain web applications without distributing and installing software on potentially thousands of client computers is a key reason for the popularity of web based applications.

The language chosen for development is PHP[3], which is a relatively new language designed for the sole purpose of creating web applications. This means that the most common tasks in developing these applications can be easily, quickly and effectively accomplished by using PHP. In addition, it is a multi-platform and non-proprietary language. A normal PHP script can be executed without changing a single line of code on any server with a PHP interpreter, i.e. Windows, Linux etc. More precisely, we have used a generic web programming PHP framework named Yii[4] for this development. Yii is an open source high-performance, component-based PHP framework for developing web applications rapidly, and

[3] http://www.php.net/
[4] http://www.yiiframework.com/

like most PHP frameworks Yii is a Model View Controller (MVC) framework. As an integrated development environment (IDE) we use NetBeans,[5] because it offers a version of the IDE specifically created for developing PHP web sites that comprises a variety of scripting and mark-up languages. Taking advantage of NetBeans IDE's support for version control and the service provided by Google code we keep our source code online at http://code.google.com/p/newsrecomender/.

4 Some selected examples

As we have said before, to represent our notion of trust we rely on the eight postulates presented in section 3.1. The implementation of the trust manager system requires translating these postulates into DeLP rules.

In order to determine if a report is trusted, distrusted or undecided the system takes as a basis the information stored in the database about users, news sources, news reports and trust judgements. This information is translated into DeLP facts, which are added to the proposed postulates to complete the trust model. Once this is done, the service is able to automatically determine which reports can be trusted by a particular user.

Suppose that Ana is a user who has logged into the system and has already expressed her trust judgements about other users and various news sources and reports. Let "google_hits_one_billion" be a news report informing that Google received 1 billion of unique visitors during May 2011. Suppose that after being translated into DeLP facts, the system's trust information is represented by the following facts:

$$
\begin{aligned}
&\mathit{report\_source}(\mathit{google\_hits\_one\_billon}, \mathit{slashdot})\\
&\mathit{report\_source}(\mathit{facebook\_hits\_one\_billon}, \mathit{etc\_news})\\
&\mathit{report\_source}(\mathit{microsoft\_hits\_one\_billon}, \mathit{msn\_news})\\
&\mathit{trust\_source}(\mathit{ana}, \mathit{slashdot})\\
&{\sim}\mathit{trust\_source}(\mathit{cristian}, \mathit{etc\_news})\\
&{\sim}\mathit{trust\_report}(\mathit{marcela}, \mathit{google\_hits\_one\_billon})\\
&\mathit{trust\_report}(\mathit{marcela}, \mathit{facebook\_hits\_one\_billon})\\
&{\sim}\mathit{trust\_report}(\mathit{marcela}, \mathit{microsoft\_hits\_one\_billon})\\
&\mathit{trust\_report}(\mathit{cristian}, \mathit{microsoft\_hits\_one\_billon})\\
&\mathit{trust\_viewer}(\mathit{ana}, \mathit{emanuel})\\
&\mathit{trust\_viewer}(\mathit{ana}, \mathit{cristian})\\
&\mathit{trust\_viewer}(\mathit{emanuel}, \mathit{marcela})
\end{aligned}
$$

Based on this and the eight postulates described earlier, the system is able to classify reports as trusted, distrusted or undecided and use this information at the moment of presenting suggestions to the user. In this case, suppose that the system needs to classify the reports "google_hits_one_billion", "facebook_hits_one_billon" and "microsoft_hits_one_billon".

In order to decide how to classify each report, the system internally tries to find a warranted argument associated with trust judgements about each of them. In figure 2 we can see that there is a warranted argument supporting the statement $\mathit{trust\_report}(\mathit{ana}, \mathit{google\_hits\_one\_billon})$, so the report should be

[5] http://netbeans.org/

trusted by Ana. In figure 3 we can see the existence of a warranted argument for $\sim trust\_report(ana, facebook\_hits\_one\_billon)$, so this report should be distrusted by Ana. Finally, figure 4 shows that the report cannot be trusted, because there is not a warranted argument for $trust\_report(ana, microsoft\_hits\_one\_billon)$. In addition, the system cannot conclude that Ana should distrust this report because it is not possible to find a warranted argument for $\sim trust\_report(ana, microsoft\_hits\_one\_billon)$. Therefore, the trust status of this report stands as undecided. With these examples we can see how trust is spread among users and what possible scenarios are possible for the epistemic status of a user regarding a specific report.

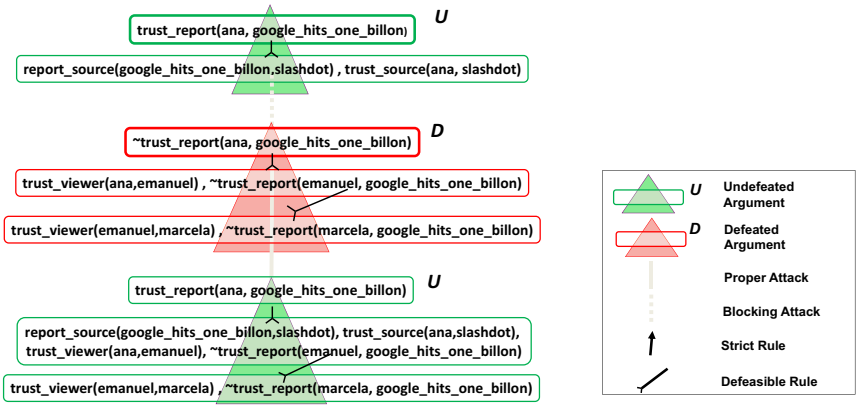

Fig. 2. DeLP dialectical tree showing the reasons to trust google_hits_one_billon.

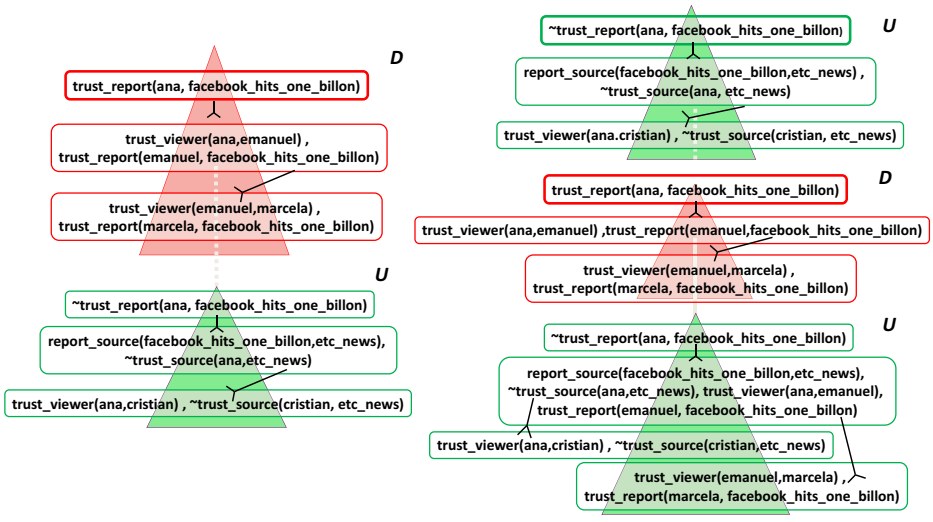

Fig. 3. DeLP dialectical tree showing the reasons to distrust facebook_hits_one_billon.

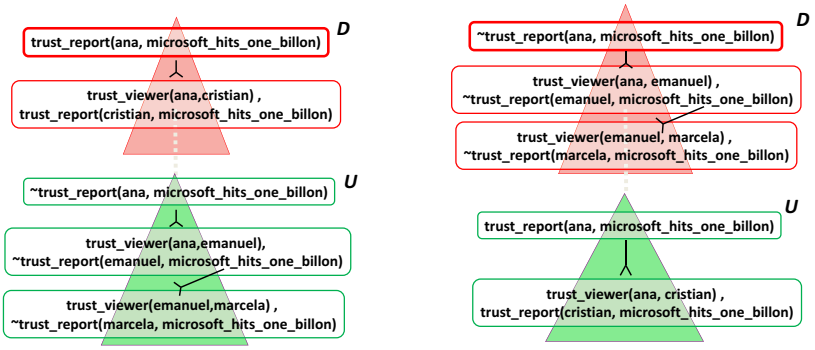

Fig. 4. DeLP dialectical tree showing no reasons to trust or distrust microsoft_hits_one_billon.

5 Conclusions and future work

We have presented the architecture of a trust-based recommendation system for news on the Web. The main contribution of this work is the description of a client-server type architecture that integrates a trust management component and a news retriever component to provide personalized recommendation to a set of end-users. The proposed architecture takes modularity as a fundamental design principle and has been designed for flexible operation with different trust managers (e.g., based on DeLP, A Prolog, etc) and to facilitate news retrieval from a variety of sources (e.g., RSS feeds, social networks, etc.). A set of examples are presented to illustrate the way in which the notion of trust can be modeled and managed by the system.

This proposal differs from previous work on news recommendation systems in allowing to draw logical conclusions about the credibility of news reports based on the opinions of a set of users. At the present time, the news management system operates and has been tested with a DeLP interpreter. This allows a convenient treatment of the defeasible nature of trust. In the future, we expect to test the system using other logics and interpreters, as well as with other mechanisms for trust propagation. In addition, we plan to integrate the tool with social networks' APIs, such as the ones provided by Facebook or Twitter, to collect large amounts of trust statements about news reports, news sources and between users. In this sense, every user will be able to contribute to and collaborate with specific communities as well as the full network of users. We anticipate that a large number of trust statements, combined with the great variety of news services currently available, have the power of revealing the full potential of qualitative approaches to news recommendation.

References

1. Budzik, J., Hammond, K.J., Birnbaum, L.: Information Access in Context. Knowledge-Based Systems 14, 37–53 (2001)
2. Carbone, M., Nielsen, M., Sassone, V.: A Formal Model for Trust in Dynamic Networks. In: Proc. of International Conference on Software Engineering and Formal Methods (SEFM'03. pp. 54–63 (2003), http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.2153
3. Chesñevar, C., Maguitman, A., Gonzlez, M.P.: Empowering recommendation technologies through argumentation. In: Argumentation in Artificial Intelligence. Springer Verlag (2010)
4. Czenko, M., Doumen, J., Etalle, S.: Trust management in p2p systems using standard tulip (January 2008), http://doc.utwente.nl/64647/
5. García, A., Simari, G.: Defeasible logic programming: An argumentative approach. Theory and Practice of Logic Programming 4(1), 95–138 (2004)
6. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 135–142. RecSys '10, ACM, New York, NY, USA (2010), http://doi.acm.org/10.1145/1864708.1864736
7. Linton, F., Joy, D., Schaefer, H.: Building user and expert models by long-term observation of application usage. In: Proceedings of the seventh international conference on User modeling. pp. 129–138. Springer-Verlag New York, Inc. (1999)
8. Nagura, R., Seki, Y., Kando, N., Aono, M.: A method of rating the credibility of news documents on the web. In: SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 683–684. ACM Press, New York, NY, USA (2006)
9. Ofuonye, E., Beatty, P., Reay, I., Dick, S., Miller, J.: How do we build trust into e-commerce web sites? IEEE Software 25, 7–9 (2008)
10. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: The Adaptive Web: Methods and Strategies of Web Personalization. pp. 325–341. Springer-Verlag (2007)
11. Sagui, F.M., Maguitman, A.G., Chesñevar, C.I., Simari, G.R.: Modeling news trust: A defeasible logic programming approach. Iberoamerican Journal of Artificial Intelligence 12(40), 63–72 (2009)
12. Sandvig, J.J., Mobasher, B., Burke, R.D.: A survey of collaborative recommendation and the robustness of model-based algorithms. IEEE Data Eng. Bull. pp. 3–13 (2008)
13. Walter, F., Battiston, S., Schweitzer, F.: A model of a trust-based recommendation system on a social network. Autonomous Agents and Multi-Agent Systems 16, 57–74 (2008), http://dx.doi.org/10.1007/s10458-007-9021-x, 10.1007/s10458-007-9021-x