

Moleskiing.it: a Trust-Aware Recommender System for Ski Mountaineering

Paolo AVESANI
Paolo MASSA
Roberto TIELLA

Recommender System (RS) suggests items to users they will like based on their past opinions. Collaborative Filtering (CF) is the most used technique and works by recommending to the active user items appreciated by similar users. However the sparseness of user profiles often prevents the computation of user similarity. Moreover CF doesn't take into account the reliability of the other users. In this paper we present a real world application, namely *moleskiing.it*, in which both of these conditions are critic to deliver personalised recommendations. A blog oriented architecture collects user experiences on ski mountaineering and their opinions on other users. Exploitation of Trust Metrics allows to present only relevant and reliable information according to the user's personal point of view of other authors trustworthiness. Differently from the notion of authority, we claim that trustworthiness is a user-centred notion that requires the computation of personalised metrics. We also present an open information exchange architecture that makes use of Semantic Web formats to guarantee interoperability between ski mountaineering communities.

- trust metric
- recommender system
- collaborative filtering
- semantic web
- website
- ski mountaineering

1

Introduction

Recommender System (RS) suggests items to users they may like, based on users' previous opinions. In general, RS assumes all the information is reliable but, in open environments, this assumption is no more true and evaluating the quality of the content provided by users becomes an important issue.

An emergent technique used to deal with the quality assessment in open environments is to ask users to explicitly specify which other users they trust. For example, on *Epinions.com*, a site where users can review products, users can also specify which other users they trust (i.e. 'reviewers whose reviews and ratings they have consistently found to be valuable' [15]) and which ones they don't. Then the user's web experience is personalised based on this 'web of trust'. Similar patterns can be found in online communities (for example, *slashdot.org* in which millions of users posts news and comments daily), in peer-to-peer networks (where peers can enter corrupted items) [1], in e-marketplace sites (such as *eBay.com*) and in general in many open publishing communities [2]. These approaches mimic real life situations in which it is common habit to rely on opinions of people we trust.

In those computational settings, Trust Metrics [3, 4, 5, 6] are emerging as a powerful technique. The idea is to use trust propagation in order to predict the level of trustworthiness in unknown users. These trust scores can then be used to personalise the user experience by emphasising content entered by trusted users and hiding content provided by unreliable ones.

In this paper, we present *moleskiing.it*: a community-based website whose goal is to make ski mountaineering safer by exploiting information and communication technologies. Users can share their opinions about the snow conditions of the different ski routes and also express how much they trust the other users. Moleskiing is a Recommender System powered with the use of trust propagation.

The contributions of this paper are as follows:

- We present a running application that is conceived to solve a real problem involving ski mountaineers.
- We argue how the use of a *local* Trust Metric can be effective in improving Recommender Systems performances and propose a preliminary efficient *local* Trust Metric.

This work is based on an earlier work 'A trust-enhanced recommender system application: Moleskiing', Proceedings of the 2005 ACM Symposium on Applied Computing (SAC'05), © 2005 ACM.
[doi.acm.org/10.1145/1067036](https://doi.org/10.1145/1067036)

- We describe Moleskiing open-publishing information architecture in which all the community information is published in Semantic Web formats. The information is not required to stay in Moleskiing servers but can be published wherever on the Web and users are anyway able to get personalised recommendations from *moleskiing.it*.

This last point is especially important from a research point of view. Research in Trust Metrics is still in its infancy and we believe one of the reasons is the lack of freely available datasets on which trying different solutions and models. The goal of Moleskiing is hence also to provide an open computational framework for studying and evaluating different Trust Metrics, Recommender Systems and in general Adaptive Personalisation techniques on data from a community of real users.

2

Ski Mountaineering domain

In ski mountaineering, both the ascent and descent of a peak are performed entirely on skis, using climbing skins and perhaps ski crampons for traction on the ascent, and then descending a continuous ski route back down to the base.

However, ski mountaineering may become very risky. Avalanches represent a ubiquitous hazard that may arise by an erroneous situation assessment.

To know in advance snow conditions plays a key role in performing a ski tour safely. However it is difficult to have first hand evidence about the snow conditions of every single route, even for security authorities. As a consequence, in order to prevent or to reduce the avalanches hazard, it is a common practice for ski mountaineers to share their experiences. Lately, it is starting to become common for ski mountaineers to publish their comments on the Web and to foster this sense of community sharing vital information. In fact, these days, a pretty common use case is the following: firstly, the ski mountaineer tries to find on the Web information about the snow conditions of some routes she would like to experience the day after, especially looking at other ski mountaineers’ diaries and reports. Secondly, she performs the routes and then, when she’s got back at home safely, she reports on the present snow conditions of the recently experienced route. In this way she contributes back with some fresh and important knowledge to the global community knowledge.

However, since we are dealing with information that can make the difference between danger and safety, there is a new, huge challenge: assessing the reliability of the ski mountaineers’ reports.

The system we present in this paper has precisely this goal: filtering ski trip annotations based on the trustworthiness of the user who entered them.

3

Trust-aware recommendations

In Recommender Systems, the standard way of filtering information in order to personalise it according to the user’s opinions is Collaborative Filtering (CF) [7, 8]. The basic idea is simple: in order to create recommendations for a certain user, CF firstly finds users whose opinions are similar to her and then recommends to her items that were liked in past by those similar users. Typically, CF is used in RSs that suggest movies, songs, books.

However, the domain of Moleskiing is somehow different from a typical CF scenario. In Moleskiing the emphasis is on security and users are invited to ‘rate’ a route also based on the present snow conditions and not only about how much they do like such a route. In this sense, users who rate routes in similar ways don’t have necessarily the same opinions about how much a route is interesting or worthwhile. Moreover, in ski mountaineering domain, the information becomes old quickly: the most important factors for security are the weather and snow conditions and of course they are not expected to remain constant over months. This fact exacerbates one of the key weaknesses of CF, data sparseness, which often makes impossible the first man-

datory step of finding users similar to the current one [9]. This problem is especially evident just after the deployment of the system when no ratings are available. Moreover, since the goal of Moleskiing is to make ski mountaineering experiences safer, a special relevance assumes the reliability of the comments entered by the users about the routes.

These peculiarities make Moleskiing a bit different from typical CF scenarios and hence we have chosen to enhance the RS with trust-aware techniques. The intuition is the following: while users can still rate routes about their current security level, they can also 'rate' other users by eliciting how much they find their comments and reports useful and accurate. This is their 'web of trust'. Then the system shows to the user mainly information provided by users she trusts. However there is a problem of coverage: every user is expected to be able to provide a direct trust statement only on a small number of other users. In order to make use also of the information provided by 'unknown' users (users the current user has not issued a trust statement on), the system exploits trust propagation. The algorithms that predict trust in unknown users based on the global trust network are called Trust Metrics and are covered in Section 5.

In this way, the cornerstones of our system become the trust statements issued by users on other users. In order to obtain this information more easily, we have adopted an open publishing architecture, in which the information (trust statements and comments on routes) can be decentralised published in Semantic Web [10] formats. As a consequence, the system can potentially aggregate the information available over every community or even single ski mountaineers' blogs and does not restrict itself to the (possibly few) users of *moleskiing.it*. A description of the open decentralised architecture is given in Section 7.

4

The Moleskiing application

In Moleskiing, there is a clear distinction between ski routes and ski trips. Ski route refers to the itinerary, while ski trip is the experience of a certain ski mountaineer at a certain date on a certain ski route. Information about ski routes are entered by an editorial board of experts because there is a good trade-off between the effort of validation process and the obsolescence rate of such data. Ski route data are mainly concerned with the starting location, the height of the mountain, the exposition of the route, and so on. These information are static and don't change. On the other hand, ski trips are concerned with a time referenced experience of a ski route. Ski mountaineers usually annotate the snow conditions and evidences or clues of potential avalanches. Data that change very quickly but that can be hardly monitored extensively. The ski mountaineering community can be conceived as a distributed network to monitor the territory. While this solution is simple and effective, the drawback is represented by the validation of data. Data collected by ski mountaineers can be partially unreliable or even totally inaccurate. A process of validation of such data is not sustainable because the obsolescence rate is greater than the certification rate. There is no break-even point in the cost of validation process.

The software architecture of the application reflects this dichotomy. The catalogue of ski routes is stored on a legacy system where a workflow supports the certification process of the editorial board. The archive of ski trips is designed as a diary and is served by a blog platform. The management of a diary is quite modular. A user can interface the application also using a third party blog server. This option is important for users that already maintain a blog somewhere. The *moleskiing.it* homepage, shown in [Figure 1](#), is therefore conceived as a twofold service: a catalogue of ski routes and a service aggregator of ski trips.

The application support an additional data entry concerned with the notion of web of trust. Just as the ski mountaineers input annotations with respect to ski routes, in a similar way they can enter annotations on other ski mountaineers. The basic idea is to allow the users to elicit information on how much they find useful or reliable the ski trip annotations of other users. In more general terms there is the opportunity to state how much a ski mountaineer is 'trustable'. From the point of view of software architecture, we will see in Section 7 how it is not required for Moleskiing users to keep this information on Moleskiing servers but they can publish it wherever on the Web, possibly in their already used blog.

The typical use-case on Moleskiing takes place as follows. A ski trip is preceded by a planning phase where a general assessment of the snow conditions is performed over a set of candidate ski routes. The choice of the destination is a trade-off between ski routes that the user likes and ski routes where information on snow conditions are available. It is straightforward to understand the role of the recent annotations of other ski mountaineers. The second phase is concerned with the outdoor performance that allows to experience for real a ski route. Eventually, the ski mountaineers update their diaries annotating with respect to the experienced ski route the up-to-date information on snow conditions.

In this scenario, the use of trust affects the application in the first step. In fact, since security is one of the goal of Moleskiing, the system tries to show to every user only reliable comments and to filter out unreliable ones.

Trust scores of users (and hence reliability of their comments) are defined on a per-user basis and hence can be different from the point of view of different users. In order to understand which users are trustworthy from the personal point of view of every single user, the system runs periodically a local Trust Metric on the overall trust network. In this way, when an user interacts with *moleskiing.it*, the system knows how much the information provided by every other user should be taken into account as useful and reliable. A description of the Trust Metric is given in [Section 5](#).

Figure 2 shows a snapshot of the application concerned with a list of recent annotated ski trips. Of course, the inverse chronological order plays a key role in the editing of this list. It is worthwhile to remember that the information on ski trips are as much important as recent they are. However quality of information is at least as important as its freshness. Quality is strong related to the reliability of ski mountaineers. To assess properly the snow conditions is not straightforward and very often requires know-how and experience. Therefore the list of most recent annotated ski trips has to be balanced with the information on their trustworthiness. In practice, only ski routes that are rated as secure in the last 15 days by the majority of 'trustable users' (users with a trust score not less than 0.6 in the [0, 1] interval) can be displayed in the list of recommended routes. These ski routes are then ranked based on the rating given by those users, weighted by their trust score. The chronological order is taken into account as well, since users reports of routes experienced recently should reflect more precisely current snow conditions.

Figure 1. Screenshot of moleskiing.it homepage

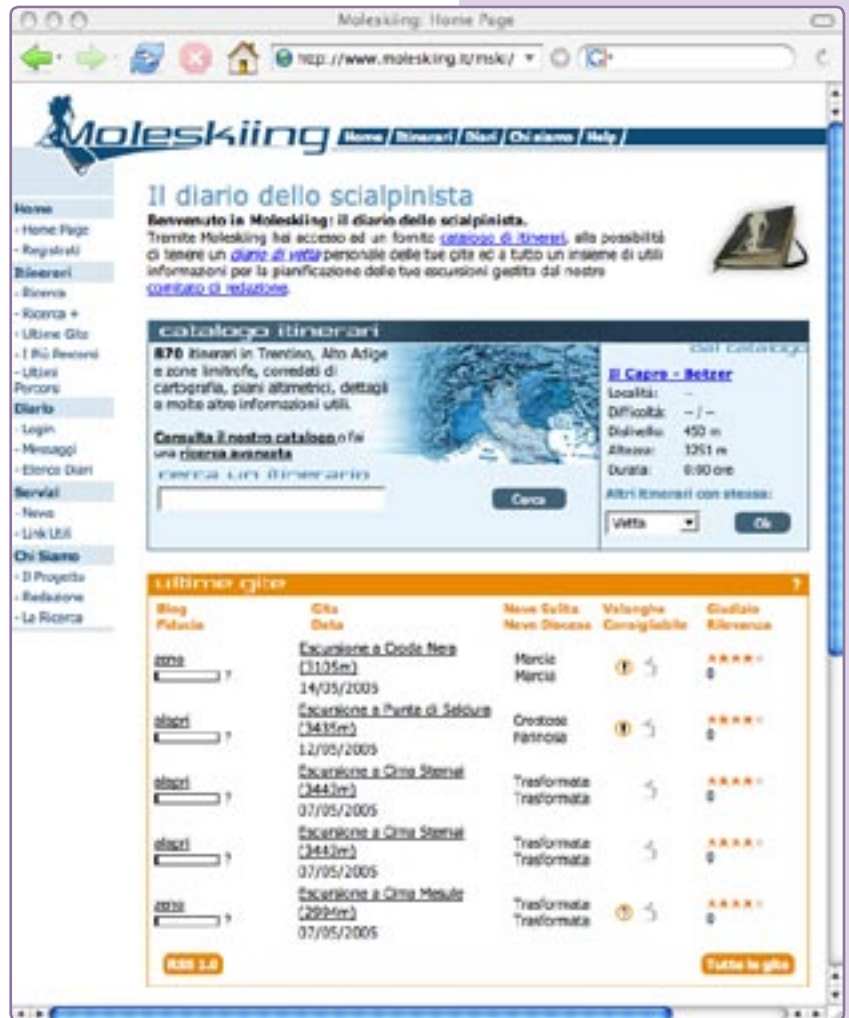


Figure 2. Snapshot of recent ski trips. The bar under a user name represents her level of trust, from the point of view of current user



The system is designed in a way that users are always invited to expand and fine tune their web of trust. In fact, it is always possible to click on a user nickname and to read her blog (containing her routes annotations and her web of trust). In this way, the current user can get a first-hand evidence and express trust in this user explicitly. Moreover, a user can always see the list of users not yet rated ordered by predicted trust or see the predicted trust score assigned by the system to every other user. Since the web experience on *Moleskiing.it* is driven by the user's web of trust, we believe it is very important that the user has it under control and is invited to update it in order to reflect her real views of other ski mountaineers' trustworthiness.

5

Trust prediction

Trust Metrics [3, 4, 5] are an emerging research topic. Currently, it is common for open publishing online communities [2] to ask users how much they find reliable the other users and then to use this information in order to give more relevance to content provided by trusted users.

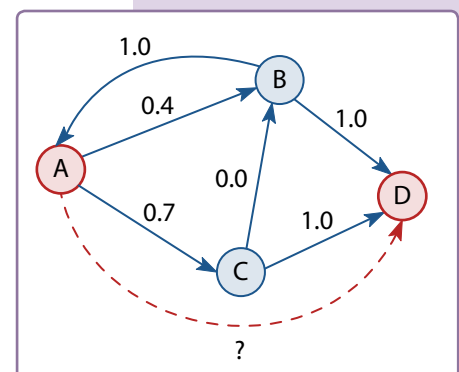
A trust network (or social network) is built aggregating all the community trust statements into a single directed weighted graph. **Figure 3** shows an example of a simple trust network. Trust statements are weighted and range from total distrust to total trust: for example, in the real interval $[0, 1]$. If there is no edge going from A to B, this means that B is unknown to A. Trust statements are also subjective: it is normal to have a user trusted with different scores by different users. They are also asymmetric in the sense that, if A trusts B as 0.9, it is totally normal that B trusts A with a different value or that B does not express trust on A, possibly because B does not 'know' A.

As already mentioned, one of the reasons for having this trust infrastructure is to provide the user with a personalised online experience, i.e. by giving more visibility to content provided by trusted users. However it is expected that the current user explicitly expresses trust only in few other users, and so the content provided by users not directly rated by current user becomes unuseful. In order to avoid this, it is possible to use Trust Metrics. Given a current user, a Trust Metric is able to predict trust scores in users she has not expressed a trust statement on, by exploiting controlled trust propagation, see **Figure 3**. The assumption is that if user A trusts B at a certain level and B trusts C at another level, something can be inferred about the level of trust of A in C.

This intuition is exploited by PageRank [11], the well known algorithm powering the search engine Google.com. The intuition is that every page on the Web has an importance value (a trust value in our terminology) and that this importance value depends on the number of the pages that link to this page and, in a recursive way, on their importance. However a drawback of this technique is that a page has the same importance value, notwithstanding the current user. This is a value that represents the opinion of the whole community about a certain user and does not provide any personalisation. We call these ones, Global Trust Metrics [9, 5].

However, by examining the real and big online community of Epinions.com users, we have found that it is often the case that different users have opposite opinions about a specific user [6]. For example, A trusts Z while B distrusts Z. In this case, we call Z a controversial user. It is self-evident that a global metric that assigns a single trust value to Z will either dissatisfy A or B or both. Basically, in presence of controversial users it is not effective to use global trust metrics. In fact, we have shown that the error produced by predicting trust in unknown users using a global metric is always higher than the error produced by a Local Trust Metric. Local Trust Metrics [9, 5] take into account the very personal and subjective views of the current user. In fact, the trust score of a certain user can be different when predicted from the point of view of different users. It can be that the Local Trust Metric suggests to A to trust Z and to B to distrust Z. In general, while Local Trust Metrics can be more precise and tailored to the single user's views, they are also computationally more expen-

Figure 3. A simple trust network. Nodes are users and direct weighted solid edges are explicit trust statement. A Trust Metric can be used to predict the dotted edge, which is missing since D is unknown to A



sive, since they must be computed for each user whereas global ones are just run once for all the community.

The previously introduced concept of controversiality is deeply intermingled with the concept of attack. In open publishing sites, it might be the case that some users have incentives to provide 'false' opinions in order to game the system. For instance, there is an interesting stream of recent research that looks into attacks to Recommender Systems [12, 13]. This issue is especially important in the domain of ski mountaineering because the reliability of users (and hence of their ski routes annotations) is one of the main concerns. For example, consider the case of a malicious user that knows a route is currently dangerous because of the snow condition but nevertheless wants to push a lot of people into going into it. This could be either in good mood because she thinks risk is what ski mountaineers look for or in bad mood because she likes to see people in danger, but from the system perspective, this is not relevant. In order to fulfil the attack, she can create many users in the system and rate with all of them as safe and pleasant the dangerous trip. This is of course a situation we want our system to handle, since its goal is to increase security. We claim that Local Trust Metrics can solve this problem. In fact, another interesting feature of local Trust Metrics is the fact they can be attack-resistant [4]: users who are considered malicious and that provide unreliable content (from a certain user's subjective point of view) can be explicitly distrusted. In this way, they are excluded from trust propagation and they don't influence the personalisation of users who don't trust them explicitly. On the other way, a global Trust Metric will probably end up taking into consideration these 'malicious' opinions and in general suffering from attacks. In the case of *moleskiing.it*, a global trust metric would easily end up recommending to every user the dangerous trip mentioned in the previous example. Since skiing security is one of the goals of the site, this aspect carries a great importance. A more complete description of Trust Metrics and related concepts can be found in [9].

Figure 4.

MoleTrust pseudocode: $\text{pred}(u)$ returns predecessors p of user u for which $\text{trust}(p) \geq 0.6$; $\text{edge}(i, u)$ is the value of the statement issued by i on u

Step 1:

```
Input: source_user, trust_net, trust_prop_horizon
dist = 0; users[dist] = source_user;
while (dist ≤ trust_prop_horizon) do
|   dist++
|   users[dist] = users reachable from users[dist - 1] and not yet visited
└   keep edges from users[dist - 1] to users[dist]
```

Step 2:

```
Output: trust_scores for users
dist = 0; trust(source_user) = 1
while (dist ≤ trust_prop_horizon) do
|   dist++
|   foreach u in users[dist]
|   └   trust(u) =  $\frac{\sum_{i \in \text{pred}(u)} \text{trust}(i) \times \text{edge}(i, u)}{\sum_{i \in \text{pred}(u)} \text{trust}(i)}$ 
```

6

MoleTrust trust metric

We have implemented a preliminary Local Trust Metric to be used in Moleskiing application. Since there are no comparative evaluations of different Trust Metrics at present time, we have chosen to start with this one and to improve it as long as real data starts to become available from users. In future, we also plan to evaluate and compare this one and other proposed metrics [3, 4, 5]. The goal of this preliminary metric is to be time-efficient so that computing trust scores in unknown users for every user does take a limited amount of time.

The trust metric, named MoleTrust was introduced in [14].

MoleTrust predicts the trust score of *source user* on *target user* by walking the social network starting from the source user and by propagating trust along trust edges. Intuitively the trust score of a user depends on the trust statements of other users on her and their trust scores. The pseudocode is presented in [Figure 4](#).

Precisely, the MoleTrust trust metric can be modelled in 2 steps. The purpose of the first step is to destroy cycles in the graph. An example of cycle is the following: *A* trusts *B* as 0.6, *B* trusts *C* as 0.8, *C* trusts *A* as 0.3. The problem created by cycles is that they require passing over a node many times adjusting progressively the temporary trust score until this value converges.

Instead we would like to have a trust metric that is able to walk on every user just once and, in doing this, to compute its definitive trust value. In this way, the running time is linear with the number of nodes.

Let us assume that we are predicting trust scores of unknown users from the point of view of user *source_user*. MoleTrust firstly orders users based on shortest-path distance from user *source_user*. In this way, it identifies all the users that are directly reachable from user *source_user*, i.e. on which user source user has expressed a trust statement. These users are at distance 1 and are called 1st degree neighbours of *source_user*. Then it identifies all the users at distance 2, i.e. users that were not rated directly by user *source_user* but there were rated by users rated by *source_user*. Going on in this way, MoleTrust divides all the users based on their distance from user *source_user*. The trust metric considers the minimum number of possible steps, so if an user is reachable in 1 step along one trust chain and in 3 steps along another trust chain, this user is a 1st degree neighbour of *source_user*.

An important parameter of MoleTrust is the trust propagation horizon: trust is not propagated at distances greater than this value. At the moment, in the Moleskiing application this value is set to 3. The intuition is that the reliability of the propagated trust decreases with every new trust step and the prediction becomes too noisy. So in the modified graph we only retain the nodes that are at distance smaller or equal to the trust propagation horizon. In this way, the number of nodes the trust metric has to consider is reduced and this means smaller computational time. Moreover, in the modified graph, only the trust statements going from users at distance *n* to users at distance *n + 1* are retained. For example, every edge from users at distance 3 to users at distance 1, 2 or 3 are removed from the social network.

The first step ends here. The social network is now a directed acyclic graph where trust flows from *source_user* to other users and it never flows back, i.e. there are no cycles.

The second step is a simple graph walk over the modified social network, starting from user *source_user* whose trust score is maximal by definition. MoleTrust computes first the trust score of all the users at distance 1, then of all the users at distance 2, etc. The trust score of one user at distance *x* only depends on trust scores of users at distance *x - 1*, that are already computed and definitive. For predicting the trust score of a user, MoleTrust analyses the incoming trust edges. However only trust edges coming from users with a predicted trust score greater than 0.6 are considered. The other users are not trustworthy and their trust statements should simply be ignored.

The predicted trust score of user *target_user* (from the point of view of *source_user*) is the average of all the incoming trust edge values, weighted by the trust score of the user who has issued this trust statement. The intuition is that opinions of users about *target_user* are weighted based on their (explicit or predicted) trustworthiness. Because of the trust horizon propagation and because of the structure of the network, it is possible that a trust metric is not able to reach every node and to predict its trust score.

At the moment, the trust predictions are run for all the users of the system as batch runs every night, since the number of users is small and this takes a reasonable amount of time. In future,

Figure 5. Moleskiing.it page where the user can express and modify her trust statements



as the number of users increases, we will use ad hoc heuristics such as, for example, predicting the trust in unknown users only for recently active users.

In this section we have presented the Local Trust Metric used in Moleskiing. We have chosen a Local Trust Metric because it can be the case that different users have different opinions about other users and we want the system to provide a personalised experience to every user, based on her beliefs and opinions.

We are aware that the first step of MoleTrust can remove significant trust links and result in inaccurate trust score predictions. This is done in order to reduce the computational time; in fact in this way there is no need to pass on one user more than once to compute her final trust value. We plan to carry an accurate evaluation and to verify if this simple but efficient trust metric is suitable enough for the Moleskiing community or if we need to design more complex trust metrics as soon as we will start collecting significant amount of data from users.

7

Open Information Architecture

Recommender Systems in general work well when they have a sizeable quantity of information available [9]. This means they require a large user base and a large number of annotations provided by those users. However, as long as new community sites arise, the potential community of users is fragmented in smaller communities, often non-exchanging information. In this way, every system can have access only to the information provided by the users of its own (often small) community. As a consequence, every system's performances tend to decrease. Moreover, a user of a community site cannot benefit of the comments of an user of another site.

A possible solution to this problem comes from the envisioned Semantic Web [10]. The Semantic Web is a project whose goal is to provide a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. In this way, it would be possible to create programs that make use of the information published in every site. Coming back to our problem, if every site involved in ski mountaineering would publish the information about its user base in some standard formats then it would be possible to create independent services, such as trust-enhanced recommender systems for instance, able to leverage the information available in every single site.

Following this intuition, we have designed an open and decentralised information exchange architecture and we have contacted other Web sites related to ski mountaineering with the goal of agreeing with a standard format suitable for publishing ski mountaineering related information, such as ski trip descriptions, user opinions about them and user opinions about other ski mountaineers. The goal is to support interoperability among medium size communities of ski mountaineers.

Figure 6. An example of FOAF file. The creator-user 'roberto' trusts 'paolo' as 9 out of 10

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<rdf:RDF xmlns="http://xmlns.com/foaf/0.1/" ...
  xmlns:trust="http://trust.mindswap.org/ont/trust.owl#">
  <Document rdf:about="http://www.moleskiing.it/blogs/roberto">
    <topic rdf:nodeID="me"/>
  </Document>

  <Person rdf:nodeID="me">
    <nick>roberto</nick>
    <weblog rdf:resource="http://www.moleskiing.it/blogs/roberto"/>
    <img rdf:resource="http://www.moleskiing.it/blogs/roberto/roberto.jpg"/>
    <knows rdf:nodeID="n1"/>
    <trust:trust9>
      <Person rdf:nodeID="n1">
        <dc:subject rdf:resource="http://www.moleskiing.it"/>
      </Person>
    </trust:trust9>
  </Person>

  <Person rdf:nodeID="n1">
    <name>roberto tiella</name>
    <weblog rdf:resource="http://www.moleskiing.it/blogs/paolo"/>
    <rdfs:seeAlso rdf:resource="http://moleskiing.it/.../paolo/foaf.rdf"/>
  </Person>
</rdf:RDF>
```

Precisely we have defined a new format (MSSA) that encodes a ski trip, an annotation made by one user about a certain route at a certain date. This information is embedded in the HTML code of the user blog page describing that trip.

The properties are *route_id* (unique identifier of the route), *trip_date* (date of the trip), *climb_snow_cond* (condition of the snow during the ascent phase), *descent_snow_cond* (condition of the snow during the descent phase), *trip_rate* (the rating assigned by the user to the route based on current snow condition), *route_rate* (the rating assigned by the user to the route based only on the quality of the route and not on current conditions), *weather_cond* (the current weather conditions), *observed_avalanches* (any presence of observed avalanches).

The most relevant property for this paper is *trip_rate*. Its value could be 0 if the user does not recommend the route based on current snow conditions and 1 otherwise. This is the most important information used by the system to decide whether a route should be recommended and shown to the user or not. As we have already said, the information provided by trusted users will have greater weight and relevance.

The other Semantic Web format we use is FOAF (Friend-Of-A-Friend) [3].

A person can encode in her FOAF file information about herself and her social relations with other persons. In particular, we use FOAF format extended with the trust extension presented in [3]. With this extension, the FOAF file creator can also express her level of trust in other users about a certain context; in our case the context is, of course, ski mountaineering. The minimum possible value is 1 and represents distrust and the maximum is 10, total trust. In a FOAF file, other users' FOAF files are identified with the *seeAlso* property and, following these links, it is possible, for example, to walk the overall ski mountaineering social network and aggregate it so that Trust Metrics can be run in order to predict trust scores in unknown users. **Figure 6** shows an example of FOAF file of a Moleskiing user and **Figure 5** shows the *moleskiing.it* interface by which a user can express and modify her trust statements.

Moleskiing exports the information about every user on the Web: the ski trip annotations in MSSA embedded in the user blog entries and the trust statements in the user FOAF file. Other ski mountaineering communities sites (*gulliver.it* and *skirando.com*, for instance) are in the process of doing the same, creating a sort of open federation. For example, a Moleskiing user will be able to express trust in a *gulliver.it* user simply by pointing to her FOAF file. Or it will be possible for a *skirando.com* user to comment a route on *moleskiing.it* site. In this way, every site beneficiaries of the user base of the other sites in order to provide better recommendations and personalisation.

However, Moleskiing does not require users to create a login on one of the federated systems in order to receive personalised recommendations and experience. For instance, bloggers can publish MSSA-extended entries in their blog and just ping Moleskiing in order to let the system know that they have created a ski trip annotation and that this should be fetched and aggregated. They can also edit their already existent FOAF file and add trust statements (related to ski mountaineering) on other ski mountaineers (pointing to their FOAF file located everywhere on the Web) as shown in **Figure 6**. Whenever they arrive on a confederated site, they can simply identify themselves by pointing to their FOAF file and blog in order to receive personalisation. While this approach requires some XML/RDF knowledge from the part of the user, the advantage is that she can keep her profile just in one single place under her control instead of having to maintain n different identities on n different sites.

8

Conclusions

Recommender Systems whose goal is to make ski mountaineering trips safer by letting users report current snow conditions of ski routes and presenting to every user only information entered by reliable users. We have argued how the ski mountaineering domain is different from standard Collaborative Filtering scenarios and how this requires a trust-enhanced approach. We have presented the preliminary local Trust Metric we use on Moleskiing and motivated the reasons behind such a choice. We have also described the open information exchange ar-

REFERENCES

1. Cornelli F., Damiani E., De Capitani di Vimercati S., Paraboschi S. and Samarati P. (2002), 'Implementing a reputation-aware gnutella server', *International Workshop on Peer-to-Peer Computing*, May.
2. Guha R. (2003), *Open Rating Systems*, Technical report, Stanford University, CA, USA.
3. Golbeck J., Hendler J. and Parsia B. (2003), 'Trust networks on the Semantic Web', *Proceedings of Cooperative Intelligent Agents*.
4. Levien R. (2003), *Advogato Trust Metric*, PhD thesis, UC Berkeley, USA.
5. 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 (EEE'04)*.
6. Massa P. and Avesani P. (2005), 'Controversial users demand local trust metrics: an experimental study on Epinions.com community', *Proceedings of the 25th AAAI Conference*.
7. Goldberg D., Nichols D., Oki B.M. and Terry D. (1992), 'Using collaborative filtering to weave an information tapestry', *Communications of the ACM*, 35(12):61-70.
8. Breese J., Heckerman D. and Kadie C. (1998), 'Empirical analysis of predictive algorithms for collaborative filtering', *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, Madison, WI, July, Morgan Kaufmann.
9. Massa P. and Avesani P. (2004), 'Trust-aware collaborative filtering for recommender systems', *Proceedings of Federated International Conference On The Move to Meaningful Internet: CoopIS, DOA, ODBASE*.
10. Berners-Lee T., Hendler J. and Lassila O. (2001), 'The semantic web', *Scientific American*.
11. Page L., Brin S., Motwani R. and Winograd T. (1998), *The PageRank citation ranking: bringing order to the web*, Technical report, Stanford Digital Library Technologies Project.
12. O'Mahony M.P., Hurley N.J. and Silvestre G.C.M. (2005), 'Recommender systems: attack types and strategies', *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-05)*, Pittsburgh, Pennsylvania, USA, 9-13 July, AAAI Press.
13. Lam S.K. and Riedl J. (2004), 'Shilling recommender systems for fun and profit', *Proceedings of the WWW'04*.

chitecture that, by using Semantic Web formats, guarantees interoperability among different ski mountaineering communities. Our future work will be concerned with the evaluation of the trust-enhanced algorithms of Moleskiing based on users' feedback. In particular, we will study if the simple but efficient proposed Trust Metric is suitable enough for the Moleskiing community and we will compare performances of different Trust Metrics on the open growing dataset.

ACKNOWLEDGEMENT

This work was partially supported by project *Montagne Sicure*, grant MIUR n.1528 del 15.05.2002 e D.D. 641 Ric. del 14.05.2002.

14. Massa P., Avesani P. and Tiella R. (2005), 'A Trust-enhanced Recommender System application: Moleskiing', *Proceedings of the ACM SAC TRECK Track*.
15. *Epinions.com* (2005), 'Web of Trust FAQ'; www.epinions.com