TrustMap: Towards Trust Recommendations for Maps

Paulo Pinheiro da Silva Nicholas Del Rio Vladik Kreinovich
Alejandro Castãneda

University of Texas at El Paso, El Paso TX 79902, USA

Abstract. The web is a rich environment for exchanging spatial information. When spatial information is shared in the form of images, i.e., maps, these images almost never come with meta-information about how they were generated. This kind of meta-information is often called knowledge provenance. Access to knowledge provenance may facilitate users to make informed decisions about the quality of maps. In this paper, we propose TrustMap, a new approach for enhancing maps with trust recommendations. For a given map, TrustMap can generate recommendations from the knowledge provenance and a network of trust relations between sources of information used to derive the map. The paper describes a TrustMap implementation as well as an on-going qualitative evaluation of the current implementation.

1 Introduction

Despite the increase of use of maps, scientists can rarely count on mechanisms allowing them to inspect the maps, understand how the maps were generated, and thus supporting their decision about accepting or not the maps as reliable artifacts. For example, in cyber-environments such as the web, maps are often shared as formated documents (e.g., PDF files), image files (e.g., PNG, JPG and GIF files), or in more sophisticated scenarios, as web applications (e.g., web mapping). Either way, maps almost never come with information that enable map users to make an informed decision about the quality of the maps that they are about to use.

This picture of how scientist share maps would not be so critical if maps were free of inaccuracies and imperfections. It is sure that most imperfections would not be critical to scientists who are using the maps to learn about some macros-aspects of a given region of interest. For other scientists, however, inaccuracies and imperfections can be very harmful to their scientific endeavor, e.g., can lead to incorrect drilling plans, unsound bridge structures, etc. The combined use of geographical information systems (GIS) and cyber-infrastructure (CI) may just aggravate the problem of assessing maps as quality scientific products since these technologies can facilitate the integration of data coming from multiple sources and multiple services. Further, maps derived from a restricted set of information sources would now rely on a multitude of people, organizations, sensors and instruments involved in the process of generating a new generation of maps.

Keeping track of information sources contributing to the generation of maps is a key activity if we want scientists to accept maps in the future. The meta-information that is used to keep track of these sources is often called *provenance information* or just *provenance*. Provenance is an important part of solutions enabling scientists to inspect maps but it is not enough to assure that most scientists would be able to better accept maps. In this paper, we propose a new approach for enhancing maps with trust recommendations. Trust recommendations is a partial and lightweight solution for the problem of accepting maps.

The rest of the paper is organized as follows. Section 2 introduces a use case that illustrates the need for trust recommendations for maps. Section 2.1 shows a gravity map and its corresponding trust map. Section 3 describes the key technologies and infrastructure supporting the generation the trust maps. With the help of the infrastructure, Section 4 describes and discuss possible methods for generating trust maps. Section 5 provides a preliminary evaluation of trust maps. Section 6 presents related work. Section 7 summarizes the main contributions of the paper.

2 TrustMap Use Case

Figure 1 show an example of a gravity contour map used here to illustrate a number of common trust issues related to maps. In this use case, scientists may use gravity data to get a rough idea of the subterranean features that exist within some region. Geoscientists are often only concerned with anomalies, or spikes in the data, which often indicate the presence of a water table or oil reserve. For example, the region in the map identified by a big oval has lower gravity readings than the rest of the map. However these anomalies have the potential to be artificial and simply imperfections introduced during the data retrieval process including for instance some data merging and filtering techniques. With the use of provenance, however, one may be able to inspect the process used to retrieve data and this figure out potential sources of imperfections []. With the use of provenance and trust, one may be able to decide whether to further investigate the are with more expensive techniques, e.g., by drilling or using controlled explosions to create a high-resolution tomography of the region.

This process of generating and accepting the map, which begins by scientists providing a region or interest or footprint, specified in terms of latitude and longitude, is defined by the sequence of tasks below:

1. Gather raw point data (possibly from multiple sources) in the region
2. Merge point data coming from distinct sources
3. Filter raw point data to remove duplicate data
4. Create a uniformly distributed dataset by applying a gridding algorithm (e.g., nearest neighbor extrapolation algorithm or the minimal curvature algorithm)
5. Create a contoured rendering of the uniformly distributed dataset

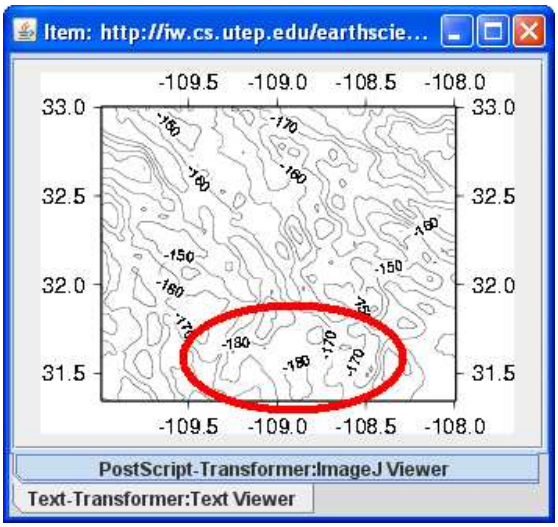

Fig. 1. Snapshot of a gravity contour map.

In a cyber-infrastructure setting, each one of the five tasks above can be realized by a web service. This set of web services is piped or chained together[1]; the output of one service would be forwarded as the input to the next service specified in the *workflow*, such as in [8].

In these types of situations where multiple workflows can satisfy a single type of request, the set of results generated by each workflow are returned to the scientist. As in any question/answer scenario, it is up to the scientist to determine what result to use. However, this situation is no different from how users interact with Web search engines. A single query often yields thousands of results, yet the burden is placed on the user to determine which answer is most appropriate. This is one of the main reasons that applications should be able to provide some trust recommendations for their results as further discussed in the following section.

2.1 TrustMap In Use

A TrustMap is an optional map that be overlaid with the map of concern an to be used to provide trust recommendations for map users. For example, Figure 2 shows a TrustMap for the gravity contour map in Figure 1. TrustMaps can be generated as map annotations in XML and be used by TrustMap browsers. In our current TrustMap implementation, we use ProbeIt! [3] not only to browse TrustMaps but also to invoke the workflow responsible for the creation of TrustMaps.

[1] This gravity map generation workflow is available for on-line use at http://iw.cs.utep.edu:8080/service-output/probeit/clientapplet.html

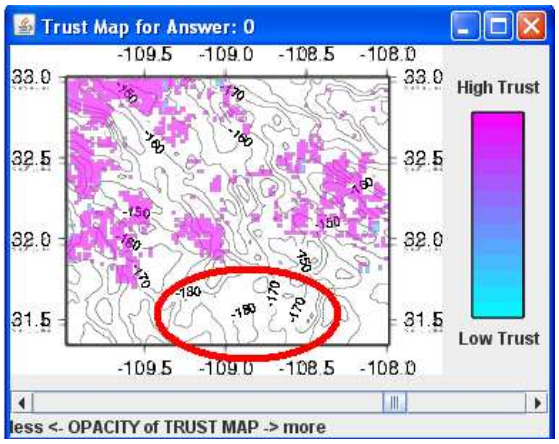

Fig. 2. TrustMap for the gravity contour map in Figure 1.

The area of interest previously identified in Figure 1 is shown in Figure 2. We can see that no trust information is available for the selected area in the TrustMap, indicating that no trust evidence could be found for the sources of the information used derive the selected part of the map. The exact interpretation of this lack of trust recommendations is very subjective in the sense that the lack of trust recommendation does not mean the presence of anything intentionally wrong (according to the current approach used to gather trust evidence, as described in Section 3.2). This lack of trust information, however, may indicate that more effort may be required for one to understand how the map was created and to eventually accept the gravity contour map as a trustworthy artifact.

Trust is a complex concepts with multiple aspects such as degrees of trust, distrust, ignorance, and inconsistencies with respect to one or multiple information sources [2]. The current TrustMap approach relies on aggregated trust values established from co-authorship between scientists. This means that TrustMaps provide information about how much a larger social network, e.g., a scientific community, may perceive and trust a given scientist and that no information about the scientist just means the lack of evidence that scientists in the social network trust the scientist. More problematic is the meaning of trust values, which in our case, are basically used for comparison. Therefore, we cannot say that source A is not trustworthy because his/her trust score is 0.321. But we can say that, with the current evidences of trust, source A may be more trustworthy than source B, that has a trust score of 0.227. With this notion of sources than can be more or less trustworthy than other, we can observe the following: *Low trust* or *no trust* – further investigation may be required before the scientist accepts the map; *High trust* – further investigation may not be required because information used for rendering areas of interest come from sources regarded as trustworthy by their social networks.

In the following sections we describe the tools, infrastructure, and methods used to generate trust maps.

3 Underlying Technologies

A trust map is kind of data layer that can be overlaid on contour maps. TrustMap datasets can be generated simply by replacing measured or derived values in a dataset with corresponding trust ratings. The process by which these trust ratings are obtained and associated with each measured or derived value are described in the following subsections.

3.1 Capturing Map Provenance

The Inference Web [9, 10] is a knowledge provenance infrastructure for conclusions derived from inference engines which supports inter-operable explanations of sources (i.e. sources published on the Web), assumptions, learned information, and answers associated with derived conclusions, that can provide users with a level of trust regarding those conclusions. The ultimate goal of the Inference Web is the same as the goal of the gravity data scenario which is to provide users with an understanding of how results are derived by providing them with an accurate account of the derivation process and the information sources used (i.e. knowledge provenance [11]).

Inference Web provides PML to encode justification information about basically any kind of response produced by agents. PML is an RDF based language defined by a rich ontology of provenance and justification concepts which describe the various elements of automatically generated proofs. Without getting into the details of the main concepts supporting PML because of obvious reasons, we can say that PML justifications are graphs with the edges always pointing towards the final justification conclusion. PML justifications can also be used to store provenance about associated information sources. PML itself is defined in OWL thus supporting the distribution of proofs throughout the Web. Each PML component, which is not yet defined here, can reside in a uniquely identified document published on the Web separately from the others.

3.2 Capturing Trust Network Meta-Information

In this paper we use an approach called CI-Learner to extract trust network information from the web. The approach starts from a given scientific community's main web portal finding scientists working for organizations affiliated to the scientific community in discourse. Scientists (people) are found through Organizations' portal(s) in the field of discourse and stored in the sets P, and O respectively. Organizations collect information about people; thus, the task is easily accomplished if we proceed from the organization to find people. Trust relationships between the scientists (i.e. set L) are determined by the co-authorship of joint publications (i.e. set D) between people we found already in set P. This

is, we query for a person's publications and keep track of publications that have two or more authors from P.

Computing the aggregate trust values for the identified networks (sets P, O, D and L, is straightforward using EigenTrust [6], we need to construct a matrix C^T using local trust relations (i.e. set L) where C_{ij} represents the number of joint publications between $person_i$ and $person_j$; and an m-vector $\mathbf{e}$ initialized to $1/m$, where m is the number of people in our network. The resulting vector $\mathbf{t}$ contains the global trust values for each member in our network.

4 TrustMap Generation

General situation: description and Monte-Carlo technique. Let us denote the data points in our gravity map by $d_1, \ldots, d_n$. These points correspond to the output of the step 4 of the process identifies in Section 2. In the probabilistic approach, we assume that we know the probabilistic degree of trust – i.e., the probabilities $p_1, \ldots, p_n$ that the corresponding data points are not outliers[2]. We want to analyze how the possibility that some of these data points are actually outliers affects the results of data processing.

A natural way to perform this analysis is to use a *Monte-Carlo approach*, i.e., to perform a large number of simulations of these outliers, and to observe how the result of data processing changes. In each simulation k, $1 \le k \le N$, we use the computer-based random number generators to keep each data point d_i with probability p_i. As a result, we get a subset $S^{(k)} \subseteq \{d_i\}$ of the original set of data points; we then apply our data processing algorithm to this subset $S^{(k)}$, and produce the result $r^{(k)}$. The values $r^{(k)}$ corresponding to different iteration provide an (approximate) description of how the possibility of outliers affects the result of data processing.

As usual in statistical analysis, we can use the mean $\bar{r} = \frac{1}{N} \cdot \sum_{k=1}^{N} r^{(k)}$ and the standard deviation $\sigma = \sqrt{\frac{1}{N} \cdot \sum_{k=1}^{N} (r^{(k)} - \bar{r})^2}$ of these values to get a good picture of the corresponding effect.

Limitations of Monte-Carlo approach. The main limitations of the Monte-Carlo approach is caused by the known fact that its accuracy is $\sim 1/\sqrt{N}$. Thus, to get an accuracy of 10%, we need to perform 100 iterations, etc. This is already a nuisance for accuracy, but for trust, this accuracy issue often makes this approach really impractical. For example, if the probability of an outlier is 1% (a very realistic number in good quality maps), then to correctly gauge the effect of

[2] The TrustMap generation process described in this section is a simplification of the actual implementation due to this assumption. In most practical cases, we may not know the degree of trust of some points in the map, as discussed in Section 2.1.

these outliers we must have $1/\sqrt{N} \ll 1\%$, i.e., we must have more than 10,000 repetitions of (already time consuming) data processing program.

It is therefore desirable to develop faster (and more accurate) algorithms for predicting the effect of trust on the results of data processing.

Case study: nearest neighbor extrapolation. In this paper, on the example of a simple map extrapolation algorithm (namely, the K nearest neighbors algorithm), we will show how the resulting trust can be estimated in reasonable time.

In this algorithm, we start with the results $v_1, \ldots, v_n$ of measuring the values of a physical field at several points $x_1, \ldots, x_n$. To estimate the value $v(x)$ of this field in a generic point x, we take K closest points $p_{i_1}, \ldots, p_{i_K}$ and take the average $\dfrac{1}{K} \cdot (v_{i_1} + \ldots + v_{i_K})$ of the corresponding values as the desired estimate.

Simplest case $K = 1$: formulas and linear-time algorithm. To show how to estimate the effect of trust on this extrapolation result, let us start with the simplest case when as an estimate for $v(x)$, we simply take the value of v at the closest point.

Let us assume that the points are already ordered, so that x_1 is the closest point to x, x_2 is second closest, etc. Under these notations, with probability p_1, the value v_1 is correct (not an outlier) and is thus taken as the desired estimate for $v(x)$. The value v_2 is taken if v_1 is an outlier (the probability of which is $1 - p_1$) and v_2 is not (the probability of this is p_2). Since different measurements are assumed to be independent, the probability $P_2^{(1)}$ of selecting v_2 as an estimate (i.e., selecting x_2 as the closest point) is equal to $(1 - p_1) \cdot p_2$. Similarly, the probability $P_k^{(1)}$ of selecting the value v_k is equal to $\pi_{k-1} \cdot p_k$, where $\pi_j \stackrel{\text{def}}{=} \prod_{i=1}^{j} (1 - p_i)$.

Computing each value $P_k^{(1)}$ requires k multiplications, so if we compute all these probabilities by simply following the above formulas, we will need a total of $1 + 2 + \ldots + n \sim n^2$ computation steps. We can actually compute all these values in linear time $O(n)$ if we:

- sequentially compute the values $\pi_0 = 1$, $\pi_{j+1} = \pi_j \cdot (1 - p_{j+1})$,
- and then compute all n values $P_k = \pi_{k-1} \cdot p_k$.

In the same linear time $O(n)$, we can compute the mean $v^{(1)} = \sum_{k=1}^{n} P_k^{(1)} \cdot v_k$, the second moment $M^{(1)} = \sum_{k=1}^{n} P_k^{(1)} \cdot v_k^2$, and the standard deviation $\sigma^{(1)} = \sqrt{M^{(1)} - (v^{(1)})^2}$.

Linear-time algorithm for $K = 2$. Let us show that similar linear-time algorithms can be proposed for $K \geq 2$. Let us start with $K = 2$. In this case, the estimate is the average of the closest and the second closest values. Thus, the

mean value of this estimate can be computed as the average of the mean value of the closest value and the mean value of the second closest value (similarly, the second moment is a similar average). We already know how to compute the mean of the closest values in linear time. Let us show that we can compute the mean of the second closest values in linear time as well.

The value v_k is the second closest if this value is correct, one of the previous values v_i ($i < k$) is also correct, and all the others are outliers. For each i, the probability of this is equal to $(1-p_1)\cdot\ldots\cdot(1-p_{i-1})\cdot p_i\cdot(1-p_{i+1})\cdot\ldots\cdot(1-p_{k-1})\cdot p_k$. In terms of our notation π_j, we can describe this probability as $\pi_{k-1}\cdot p_k\cdot o_i$, where we denoted $o_i \stackrel{\text{def}}{=} p_i/(1-p_i)$ – the "odds" corresponding to the probability p_i. Thus, the overall probability $P_k^{(2)}$ that x_k is the second closest point is equal to the sum of these probabilities corresponding to $i = 1, 2, \ldots, k-1$, i.e., to $P_k^{(2)} = \pi_{k-1}\cdot p_k\cdot s_{k-1}^{(1)}$, where $s_j^{(1)} \stackrel{\text{def}}{=} \sum_{i=1}^{j} o_j$.

Similarly to the case $K = 1$, the literal use of this formula requires time $O(n^2)$, but all these probabilities can be computed in linear time if we:

- sequentially compute the values $\pi_0 = 1$, $s_0^{(1)} = 0$, $\pi_{j+1} = \pi_j\cdot(1-p_{j+1})$, $s_{j+1}^{(1)} = s_j^{(1)} + p_{j+1}/(1-p_{j+1})$,
- and then compute all n values $P_k^{(2)} = \pi_{k-1}\cdot p_k\cdot s_{k-1}^{(1)}$.

In the same linear time $O(n)$, we can compute the mean, the second moment, and the standard deviation – a scalar characteristic of the effect of trust on the estimated value $v(x)$.

Linear-time algorithm for $K = 3$. For $K = 3$, we estimate $v(x)$ is the average of the values of v in the closest, second closest, and the third closest points. Thus, to find this estimate (and to find the mean and the standard deviation of this estimate), we must be able to compute the probability $P_k^{(3)}$ that x_k is the third closest point. Similar to the case $K = 2$, we conclude that $P_k^{(3)} = \pi_{k-1}\cdot p_k\cdot s_{k-1}^{(2)}$, where $s_j^{(3)} \stackrel{\text{def}}{=} \sum_{i_1<i_2\le j} o_{i_1}\cdot o_{i_2}$.

Here, the literal use of this formula require time $O(n^2)$ for computing each sum $s_j^{(3)}$ and thus, an even larger overall time $O(n^3)$. However, it is possible to compute all these probabilities in linear time. Indeed, once can easily check that the only new pairs ($i_1 < i_2$) added when we increase j by 1 are the pairs in which $i_2 = j+1$. Thus, the difference $\Delta s_{j+1}^{(2)} \stackrel{\text{def}}{=} s_{j+1}^{(2)} - s_j^{(2)}$ has the form $o_{j+1}\cdot\sum_{i=1}^{j} o_i$, i.e., the form $\Delta s_{j+1}^{(2)} = o_{j+1}\cdot s_j^{(1)}$. So, all the probabilities $P_k^{(3)}$ can be computed in linear time if we:

- first sequentially compute the values $\pi_0 = 1$, $s_0^{(1)} = 0$, $\pi_{j+1} = \pi_j\cdot(1-p_{j+1})$, $s_{j+1}^{(1)} = s_j^{(1)} + p_{j+1}/(1-p_{j+1})$,
- then $s_0^{(2)} = 0$, $s_{j+1}^{(2)} = s_j^{(2)} + o_{j+1}\cdot s_j^{(1)}$,

– and finally, all n values $P_k^{(3)} = \pi_{k-1} \cdot p_k \cdot s_{k-1}^{(2)}$.

Linear-time algorithm for a general K. For an arbitrary K, we estimate $v(x)$ is the average of the values of v in the closest, second closest, ..., and the K-th closest points. Thus, to find this estimate (and to find the mean and the standard deviation of this estimate), we must be able to compute the probability $P_k^{(J)}$, $J = 1, \ldots, K$, that x_k is the J-th closest point. Similar to the case $K = 2$, we conclude that $P_k^{(J)} = \pi_{k-1} \cdot p_k \cdot s_{k-1}^{(J-1)}$, where $s_j^{(J-1)} \stackrel{\text{def}}{=} \sum_{i_1 < \ldots < i_{J-1} \le j} o_{i_1} \cdot \ldots \cdot o_{i_{J-1}}$. Here, $s_{j+1}^{(J)} = s_j^{(J)} + o_{j+1} \cdot s_j^{(J-1)}$. Thus, all the probabilities $P_k^{(J)}$ can be computed in linear time if we:

– first sequentially compute the values $\pi_0 = 1$, $s_0^{(1)} = 0$, $\pi_{j+1} = \pi_j \cdot (1 - p_{j+1})$, $s_{j+1}^{(1)} = s_j^{(1)} + p_{j+1}/(1 - p_{j+1})$,
– then the values $s_0^{(2)} = 0$, $s_{j+1}^{(2)} = s_j^{(2)} + o_{j+1} \cdot s_j^{(1)}$,
– then the values $s_0^{(3)} = 0$, $s_{j+1}^{(3)} = s_j^{(3)} + o_{j+1} \cdot s_j^{(2)}$,
– ...
– then the values $s_{j+1}^{(J)} = s_j^{(J)} + o_{j+1} \cdot s_j^{(J-1)}$,
– ...
– then the values $s_{j+1}^{(K-1)} = s_j^{(K-1)} + o_{j+1} \cdot s_j^{(K-2)}$,
– and finally, all n values $P_k^{(J)} = \pi_{k-1} \cdot p_k \cdot s_{k-1}^{(J-1)}$ for all k and $J = 1, \ldots, K$.

There are K linear-time steps, so, for all K, we can indeed compute the effect of trust in linear time $O(n)$.

Comment. The exact formulas require that we consider all n map points. However, the probability that the k-th point is the closest is proportional to the probability π_{k-1} of k small values $1 - p_i$. Even if each of these values is 10%, very fast the product becomes negligibly small. Thus, in practice, to find the effects on $v(x)$, we only need to consider a few neighboring points of this point x.

5 Preliminary TrustMap Evaluation

5.1 Evaluation Procedure

In the context of our study we define "acceptable" maps as artifacts maintaining the following properties: the source data used to derive the map has no known imperfections; any data aggregations or merges were performed correctly according to specifications provided by domain experts in the field (e.g., eliminating duplicate data or ensuring that data is semantically and syntactically compatible); all services used to generate the map are semantically compatible according to domain experts (for example, gridding or interpolation services are suitable for the type and density distribution of the source data); must have a high resolution; and must have high coverage (sufficient data points for the size and resolution of the map).

Our experiment required subjects to classify maps as "acceptable" or "unacceptable" given our definition, and to provide a short explanation justifying their answer. As with any experiment, where the test data is generated, it is important to provide subjects with realistic scenarios (i.e., test data that was generated following some distribution that makes sense for the subject matter). In our case, it was important to provide subjects with a balanced set of gravity contour maps, meaning we considered each kind of map violation, number of sources used to derive the map, and region of map to be dimensions that needed to be balanced (i.e., equally likely factors).

Given our balanced set of test maps, each subject was asked to classify each map as either "acceptable" or "unacceptable", sometimes having the TrustMap visualization enabled and sometimes not and to provide an explanation for their decision. From this perspective, subjects play two roles; as users with both the test maps and TrustMap visualization denoted as group *map + tm* and as users with only the test maps denoted as *map*. Our goal is to justify the anticipated higher performance of group *map + tm* in classifying maps, with a sound statistical support.

The entire experimental process, including the selection of subject test maps, the presentation of the test maps, the enabling and disabling of TrustMap features, and the recording of results are all automated and implemented as Java applets thus allowing subjects to participate in the experiment remotely and at their own leisure.

5.2 Results

As of now, we have only a limited number of subjects and are thus unable to statistically justify our current findings. One of the user actions that are recorded in our experiment, however, is whether or not the TrustMap feature was accessed by a subject. This gives us some qualitative insight as to whether users "like" to use the TrustMap feature, in which case they did as every subject thus far used the feature every time to help them to decide *what* to classify each map as. Additionally, subjects usually supported their decisions using only the information visualized by TrustMap.

Although it seems apparent that users prefer to use TrustMap, we cannot yet say that TrustMap makes them better at classifying maps as trustworthy artifacts at this time.

6 Related Work and Discussion

The uses of provenance and trust are dictated by the goals of the particular systems; because various dimensions of provenance can be used to achieve various goals, there is no one use fits all. For instance, a first category of provenance systems aim at providing users with a sort of "history of changes" associated with some workflow, thus their view of provenance differs from that of a second category of provenance systems, which aim at providing provenance for use of

debugging, or understanding an unexpected result. A third category of provenance systems record events that are well suited for re-executing the workflow it is derived from. From this point of view, the TrustMap approach is more appropriate for the second category of provenance systems.

VisTrails, a provenance and data exploration system, provides an infrastructure for systematically capturing provenance related to the evolution of a workflow [4]. VisTrails users edit workflows while the system records the various modifications being applied. In the context of this system, provenance information refers to the modifications or history of changes made to particular workflow in order to derive a new workflow; modifications include, adding, deleting or replacing workflow processes. VisTrails provides a novel way to render this history of modifications. A treelike structure provides a representation for provenance where nodes represent a version of some workflow and edges represent the modification applied to a workflow.

MyGrid, from the e-science initiative, tracks data and process provenance of workflow executions. Authors of MyGrid draw an analogy between the type of provenance they record for in-silico experiments and the kind of information that a scientist records in a notebook describing where, how and why experimental results were generated [14]. From these recordings, scientists are able to operate in three basic modes: (i) debug, (ii) check validity, and (iii) update mode, which refer to situations when, a result is of low quality and the source of error must be identified, when a result is novel and must be verified for correctness, or when a workflow has been updated and its previous versions are requested. Currently, MyGrid RDF provenance is viewed using Haystack [12]. Haystack displays the provenance log as a labeled directed graph tailored to the needs of a specific user; only relevant provenance elements related to the role of the specific user browsing the provenance are rendered. In this scenario, connections between different resources are rendered allowing users to realize the relationships between provenance elements such as inputs/outputs and applied processes and thus realize the execution trace.

The Earth Science System Workbench (ESSW) is another effort at capturing and presenting scientific results to users [5]. Upon user requests, ESSW leverages a suite of Notebook tools that can display both the scientific result and the associated provenance. Stored scientific visualizations such as swaths [5] are rendered in HTML upon request; the request is in the form of a query. Additionally, ESSW leverages GraphViz [7] in order to graphically render the execution trace in the form of a directed graph, where nodes are data objects and edges define relationships between objects, similarly to how Probe-It! renders justifications.

Karma [13] is a non-obtrusive provenance recorder for scientific results from Indiana University. Karma, unlike ESSW provides an in-house approach for rendering provenance; an algorithm accurately pieces together a directed acyclic graph that describes the data or process provenance. Karma is primarily targeted at capturing provenance associated with service oriented workflows, thus rich provenance associated with Web service invocations are captured by the system.

On the commercial side, ArcGIS from ESRI allows users to both develop and execute workflows (or "models" as called by ArcGIS). From a workflow, users have access to the final result, i.e., a map, intermediate results, and meta-data associated with the source data. Additionally, all these elements associated with a model can be visualized. ArcGIS tools draw no distinction between executable models and execution traces of models; no view of a model's execution trace is provided, only the model itself. Therefore, ArcGIS may not necessarily support provenance visualization and trust recommendations as defined in this paper. However the model provides certain features such as data point visualization which can be used to analyze final results and thus identify and explain map imperfections.

This study on provenance and trust is an attempt to verify the effectiveness of new methods for quality assessment other than the more traditional approaches such as uncertainty propagation or error-model development, which we believe are complimentary to provenance visualization. In 1991, the National Center for Geographic Information and Analysis (NCGIA), held a four day meeting, in which GIS and spatial data expert met and tried to come to a consensus on the definition spatial quality and the different factors that contribute to it. Additionally, the participants expressed their thoughts on how quality could be visualized; some of the participants even suggested that users of GIS systems, as well as being able to visualize target data, should also be provided with visualizations of the error model associated with that data. However, the majority felt that when applicable, datasets should be visualized as some graphic or image rather than in its raw tabular for, thus enabling researchers to identify aberrant data more rapidly [1]. The discussion at that time did not appear to move into a more complex notion of trust as presented in this paper.

7 Conclusions

This paper described the TrustMap approach for trust recommendations in support of a map generated from a real end-to-end, cyber-infrastructure based application. The paper described the techniques and infrastructure required for the generation of trust recommendations. Requirements include the use of map provenance information encoded in the Proof markup Language to identify the multiples sources of information used to derive the map; the use of CI-Learner to extract social networks from scientific portals that include people and organization identified as information source for the map; and the use of EigenTurst to compute aggregated trust values for the sources. With the identification of the map provenance, sources, and degree of trust on the sources, the paper described some approaches for computing trust recommendations. The evaluation done so far has demonstrated that most subjects felt more comfortable using maps with trust recommendations than ordinary maps without trust recommendations.

Unfortunately, at this time, we are unable to claim anything about the effects of TrustMap in scientists' cognitive processes of accepting maps other than the initial impressions reported in this paper. As our study and evaluation data ma-

ture, however, we believe that both expected and unexpected TrustMap benefits will be revealed.

References

1. K. Beard, Barbara, Buttenfield, and W. Mackaness. Visualization of the Quality of Spatial Information: Closing Report. Technical report, University of Maine and University of New York, 1994.
2. M. D. Cock and P. Pinheiro da Silva. A Many Valued Representation and Propagation of Trust and Distrust. In *In Proceedings of International Workshop on Fuzzy Logic and Applications (WILF2005)*, pages 108–113, Crema, Italy, 2006. Springer.
3. N. Del Rio and P. Pinheiro da Silva. Probe-it! visualization support for provenance. In *Proceedings of the Second International Symposium on Visual Computing (ISVC 2)*, pages 732–741, Lake Tahoe, NV, 2007. Springer.
4. J. Freire, C. T. Silva, S. P. Callahan, E. Santos, C. E. Scheidegger, and H. T. Vo. Managing Rapidly-Evolving Scientific Workflows. In *Proceedings of the International Provenance and Annotation Workshop (IPAW)*, 2006. (to appear).
5. J. Frew and R. Bose. Earth System Science Workbench: A Data Management Infrastructure for Earth Science Products. In *Proceedings of the 13th International Conference on Scientific and Statistical Database Management*, pages 180–189, Fairfax, VA, July 2001.
6. S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, 2003.
7. A. R. Labs. AT&T Graphiz. http://www.graphviz.com.
8. B. Ludaščher and et al. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice & Experience*, 2005. Special Issue on Scientific Workflows.
9. D. L. McGuinness and P. Pinheiro da Silva. Infrastructure for Web Explanations. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *Proceedings of 2nd International Semantic Web Conference (ISWC2003)*, LNCS-2870, pages 113–129, Sanibel, FL, USA, October 2003. Springer.
10. D. L. McGuinness and P. Pinheiro da Silva. Explaining Answers from the Semantic Web. *Journal of Web Semantics*, 1(4):397–413, October 2004.
11. P. Pinheiro da Silva, D. L. McGuinness, and R. McCool. Knowledge Provenance Infrastructure. *IEEE Data Engineering Bulletin*, 25(2):179–227, December 2003.
12. D. Quan, D. Huynh, and D. Karger. Haystack: A platform for authoring end user semantic Web applications. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 738–753, 2003.
13. Y. L. Simmhan, B. Pale, and D. Gannon. A Survey of Data Provenance Techniques. Technical Report IUB-CS-TR618, Computer Science Department, Indiana University, USA, 2005.
14. J. Zhao, C. Wroe, C. Goble, R. S. andq D. Quan, and M. Greenweed. Using Semantic Web Technologies for Representing E-science Provenance. In *Proceedings of the 3rd International Semantic Web Conference*, pages 92–106, November 2004.

This article was processed using the LaTeX macro package with LLNCS style