Establishing Workflow Trust Using Provenance Information

Shrija Rajbhandari, Arnaud Contes, Omer F.Rana, Vikas Deora, and Ian Wootten

School of Computer Science, Cardiff University,
Queen's Buildings, 5 The Parade, Cardiff CF24 3AA, UK

Abstract. Workflow forms a key part of many existing Service Oriented applications, involving the integration of services that may be made available at distributed sites. It is possible to distinguish between an "abstract" workflow description outlining which services must be involved in a workflow execution and a "physical" workflow description outlining the instances of services that were used in a particular enactment. Provenance information provides a useful way to capture the physical workflow description automatically - especially if this information is captured in a standard format. Subsequent analysis on this provenance information may be used to evaluate whether the abstract workflow description has been adhered to, and to enable a user executing a workflow-based application to establish "trust" in the outcome.
An analysis tool that makes use of provenance information to assist in evaluating trust in the outcome of a workflow execution is presented. The analysis tool makes use of a rule-based engine, supporting a range of queries on the recorded information by one or more workflow enactors. The results of the analysis tool on a particular workflow scenario are presented, along with an experiment demonstrating how the analysis tool would scale as the granularity of the recorded provenance information was increased.

1 Introduction

Computational scientists in recent years have been increasingly relying on distributed computing technologies as an essential part of their everyday research. Although the concept of sharing distributed resources amongst geographically distributed groups is not new, increasing advancement in Service Oriented Architectures (SOA) in Grid and Web Services makes the vision more realistic. Amongst the consequences of the progress toward SOA in scientific domain is an increased emphasis on provenance data, and the need for mechanisms to acquire, use and manage such data. Workflow forms a key part of many existing Service Oriented applications, to integrate services that may be made available at distributed sites. It is possible to distinguish between an"abstract" workflow description outlining which services must be involved in a workflow execution and a "physical" workflow description outlining the particular instances of

services that were used in a particular enactment. Provenance information provides a useful way to capture the physical workflow description automatically especially if this information is captured in a standard format [1]. Subsequent analysis on this provenance information may be used to evaluate whether the abstract workflow description has been adhered to, and to enable a user executing a workflow-based application to establish "trust" in the outcome of the physical workflow.

Our work aims to assess the "trust" that a user can place in the result that has been produced as an outcome of a workflow execution. Trust in this context is defined as the ability of a service to perform as advertised. Such assessment can be particularly useful if a user wishes to automatically compose a workflow in the future, or determine which sets of services could be successfully combined. Consequently, trust assessment can only be achieved effectively *after* a service has been provisioned – as it is then possible to compare the advertised service description with what was actually provided. We envision the development of an autonomic workflow enactment engine that can automatically chose between a set of discovered services, and combine them to produce an application. The approach presented here therefore adds to the existing literature on providing semantic annotations to services to enable their composition.

Providing provenance information along with the result (as an outcome of a workflow) improves a user's ability to judge the validity of the result. Although provenance provides justification for the result, the notion of how much trust can be placed in the result is completely implicit – to the extent that such concern has not been fully addressed in existing workflow systems. This paper introduces an analysis tool that makes use of provenance to assist in the "trust assessment" of the result that has been produced through a distributed workflow session. The analysis tool makes use of a rule-based engine, supporting a range of queries on the recorded provenance information by one or more workflow enactors. The analysis tool provides information that is consumed to elicit/attain some measure of "result trustworthiness".

Related work on trust models is presented in section 2. Section 3 presents our workflow trust architecture. Section 4 provides the rule-based analysis tool that is the basis for our trust architecture, and section 5 provides an evaluation of our model with the workflow scenario from the BioDiversityWorld(BDW) project [2].

2 Related Work

Significant literature exists for calculating "trust" based on the reputation or Quality of Service (QoS) of actors [3–6]. Such approaches achieve trust evaluation in two parts. First, to allow actors to trust each other there is a need to endow them with the ability to reason about the reliability, or honesty of their counterparts. This ability is captured through trust models. Second, to enable actors to calculate the degree of trust they can place in their interaction partners. A high degree of trust in an actor would mean it is likely to be chosen as an interaction partner. Hence, trust models aim to guide an agent in deciding

how, when, and who to interact with. However, in order to do so, trust models initially require actors to gather some knowledge about the characteristics of their counterparts. Based on existing work, this may be achieved as follows:

1. **A presumption drawn from the actor's own experience:** Trust is computed as a rating of the level of performance of the actor. The actor's performance is assessed over multiple interactions checking how good and consistent it is at doing what it says it does. To this end, Witkowski et al. [7] propose a model whereby the trust in an actor is calculated based on its performance in past interactions. Similarly, Sabater et al. [4] propose a similar model but do not just limit the overall performance to the actor's direct perception, but they also evaluate its behavior with other actors in the system.
2. **Information gathered from other actors:** Trust in this approach is drawn indirectly from recommendations provided by others. As the recommendations could be unreliable, the actor must be able to reason about the recommendations gathered from the other actors. The latter is achieved in different ways: (1) deploying rules to enable the actors to decide which other actors' recommendation they trust more [8]; (2) weighting the recommendation by the trust the actor has in the recommender—EigenTrust [3] and PageRank [9] are examples of this approach.
3. **Socio-Cognitive Trust:** Trust here is drawn by characterizing the known motivations of the other actors. This involves forming coherent beliefs about different characteristics of these actors and reasoning about these beliefs in order to decide how much trust should be put in them [10]. Deriving such motivations in a real world system is a non-trivial task, and this level of trust is therefore often difficult to compute in practice.

Refer to [11] for more details on trust and reputation approaches. The aim of such existing work is to help in the automatic selection of a trustworthy actor based on the evaluated trust for each actor. Other approaches, such as [12–14], involve trust assessment for service composition. The primary objective within these approaches is to: (1) select trustworthy services to achieve a particular activity; (2) compute optimal execution plans for a workflow. Our framework differs from such models as the concern is towards trustworthiness of an outcome that is the result of a scientific experiment—performed in a distributed, service oriented environment. To achieve this we recognize the importance of provenance data and exploit this in our trust architecture. Thus, apart from provenance data providing the explanation about how a result has been produced, it also provides a way to formulate the trustworthiness that one can place in the result. A trust assessment framework consisting of an analysis tool that makes use of provenance information to assist in evaluating trust in the outcome of a workflow execution is presented.

Automatically inferring such trust using provenance information leads to workflow systems that can adapt service selection and provisioning over time. For instance, consider a scenario where an abstract workflow has been developed

by a user describing the types of services they wish to compose. Each service in this instance is a place holder (proxy), that is resolved to a given service instance during the enactment process. Trust metrics therefore enable the enactment engine to dynamically bind a proxy to a more trustworthy instance of a service (if multiple instances can be found), thereby leading to "autonomic" workflow enactment mechanisms.

3 Workflow Trust Architecture

The trust architecture in figure 1 consists of two main parts: (1) a trust calculator; and (2) an analysis tool. "Trust assessment" via the trust calculator is achieved by performing rule-based analysis on provenance data that has been recorded about a workflow. The necessary provenance data describing the "physical" workflow that has been enacted is encoded in a standard XML schema and recorded within a Provenance Store [1].

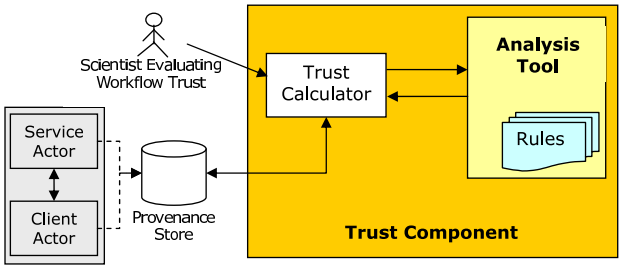

Fig. 1. Trust Architecture

Provenance information utilized for trust evaluation may be categorized as follows: (1) Process Provenance: corresponds to the steps involved in the workflow that lead to a result. It also includes the inputs and outputs for each service involved in the process; (2) Actor State Provenance: records the state of the actors/services involved in a particular workflow instance. This also includes static data such as ownership and identity information associated with an actor.

3.1 Trust Calculator

The trust calculator allows users to pose queries to the Provenance Store for retrieving provenance information associated with previous workflow executions. A user may query data for each stage of the workflow. The trust calculator adopts a decision process for analyzing the queried provenance. The conceptual decision process is presented in [15], where a decision tree model has been used to generate trust measure for a workflow result. Each node within the decision tree corresponds to a question which triggers an analysis on previously recorded

data associated with the workflow. The response to a question is a Boolean value – for example, a result from checking for "no-conflict" on data passed between two services in the workflow can be either (1) "True" (positive) if there is no conflict or (2) "False"(negative) if a conflict exists. In [15], the questions are processed only through user intervention. However, our current work involves mapping the questions in the decision tree to rules in the analysis tool, so that the answer to a particular question can be evaluated automatically. The analysis tool is discussed in section 4 with some example rules to illustrate the concepts.

We adopt a beta probability distribution for combining analysis results and for expressing trust measures. This distribution is useful for modelling random probabilities and proportions, particularly in the context of Bayesian analysis. In particular, the Bayesian theory uses standard beta distributions to model posterior probability estimates of observed binary events with two possible outcomes. The mathematical analysis that leads to the posterior probability estimates of binary events can be found in Bernardo and Smith [16]. In our case, the binary events are the different analysis performed by the analysis tool with Boolean outcomes.

We choose the beta density function that takes the integer number of these two possible outcomes represented as parameters α and β (representing the total number of positive and negative outcomes respectively) to express the uncertain probability that the workflow result may be trusted.

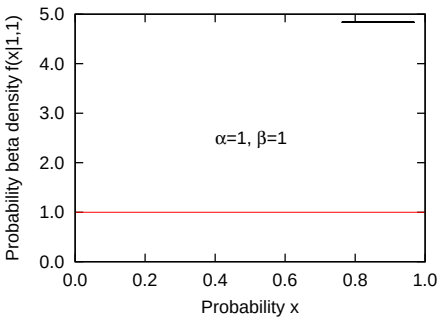

Fig. 2. Uniform distr. $\alpha = 1, \beta = 1$

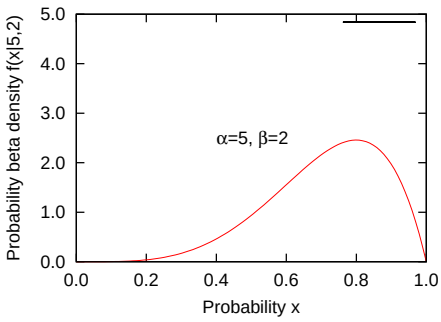

Fig. 3. Example distr. $\alpha = 5, \beta = 2$

The Beta Density Function: The Beta Distribution is a continuous probability distribution with the probability density function defined on the interval [0, 1]. Beta distribution is defined in terms of parameters α and β. A continuous random variable has a beta distribution with parameters α and β, its density function $f(x|\alpha, \beta)$ can be expressed as;

$$f(x|\alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B[\alpha, \beta]} \tag{1}$$

where, $0 \leq x \leq 1$, $\alpha > 0, \beta > 0$ and $B[\alpha, \beta]$ is the beta function with parameter α and β which is given as;

$$B[\alpha, \beta] = \int_0^1 x^{\alpha-1}(x-1)^{\beta-1}dx \tag{2}$$

A special case in beta distribution is when $\alpha = 1$ and $\beta = 1$, x is said to have a uniform beta distribution. Thus when nothing is known, i.e., no analysis is performed on the workflow, the distribution is uniform as seen in Fig.2. Let us consider an analysis process of two possible outcomes $\{positive, negative\}$, let p be the total number of observed *positive* outcomes and n be the total number of observed *negative* outcomes. After this observation, the posterior distribution is the beta function with $\alpha = p+1$ and $\beta = n+1$. Figure 3 illustrates an example of the beta density function $f(x|\alpha, \beta)$ with 4 positive and 1 negative outcomes. This beta density function distribution is a way to express the uncertain probability that a process will produce a positive outcome in the future [15]. This provides a firm mathematical basis for combining the responses from the analysis tool and expressing trust measures for the results of past workflows under evaluation. The probability distribution of x is continuous, so it is only meaningful to compute $f(x|\alpha, \beta)$ for a specific interval$[0, 1]$. We consider the maximum point in the distribution of x to be the "trust probability" of the result given the amounts of positive and negative outcomes from the analysis. In the example in Figure 2 with 4 positive and 1 negative outcomes, the "trust probability" is 0.8. This indicates that given the number of analysis outcomes, the probability to trust the result is 0.8. In a case with 5 positive and 0 negative outcomes, the trust probability is approx. 1, representing complete trust in the result.

4 Analyzing Workflow to Elicit Trust

As stated previously in section 3, the assessment of the amount of trust for a particular result is obtained by processing the decision tree. In [15], these questions were manually answered by a user. In this section, we describe the underlying mechanism used to automate this process.

4.1 Analysis Tool

Our analysis tool makes uses of the Java Expert System Shell (`JESS`), a java rule engine. `JESS` uses an enhanced version of the `Rete` algorithm to process rules. `Rete` is an efficient mechanism for solving the difficult many-to-many matching problem (see for example [17]). The `Rete` algorithm expects two different types of input, (1) a set of *rules* which represent the logic of the computation (also called *production rules*) and (2) a set of facts which represent the data to be analysed (also called *working memory*).

The data produced by the execution of a provenance-aware workflow is composed of a set of *p-assertions*. Such a set of p-assertions provide the description of the workflow that has actually been enacted (an instance of an abstract workflow description). A p-assertion can be used to record one of the following events: an interaction between two actors, the state of an actor at a particular moment or a relation between two events.The analysis tool can also be used to detect possible conflicts in the p-assertions recorded. The nature of detected conflicts is large and various, from detecting a difference between the data submitted by the sender and by the receiver of a given interaction, to the detection of an unexpected behaviour during the execution of a workflow.

In the current implementation, p-assertions are provided under an `XML` format, defined by a particular schema referred as a `PStructure` [18]. A p-assertion is composed of two kinds of data: the first describes provenance-specific data such as client (requester) and service (provider) actors, unique identifier for the p-assertion and the second represents application-specific data. This application-specific data is also encoding in an `XML` format but defined by external, application specific, schemas. Therefore, the analysis tool has to manipulate data in which a part of the structure (and the meaning) is known (the provenance-specific part) and another part which could vary according to the particular application being considered.

Our analysis tool avoids the need for a dedicated `XML` parser, and is able to import any kind of `XML` content. It is also, subsequently, able to perform some reasoning on this data. The architecture of the analysis tool, shown in Figure 4, makes use of three components: the *XML Loaded*, the *Facts Transformer* and the *Facts Processor*.

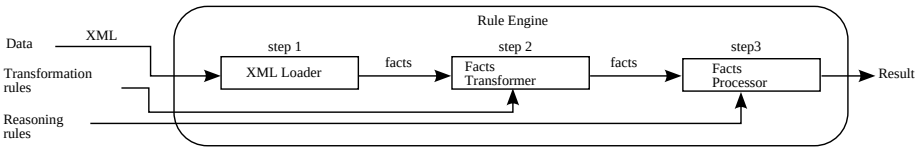

Fig. 4. Architecture of the analysis tool

The processing of data is composed of three successive steps involving the three components introduced previously:

- *Step 1 - Populating the Rule Engine*: the *XML loader* is in charge of converting the `XML` into a set of *generic facts*. We have defined two generic templates to map the `XML` structure into a set of facts. The first, called `Element`, represents an `XML` element. The second, called `Attribute`, represents an attribute associated with a particular `XML` element. Each element of the `XML` document is loaded into the memory of the rule engine as a *fact* called *Element*. Each `Element` is identified by a unique identifier `ElementID`. An `Attribute` shares the same `ElementID` that the `Element` belongs to. The relation(s) between

elements (eg: parent, sibling, sub-element) are enforced by inserting in each `Element` the list of its parent, children and attributes, if any. Based only on these two templates, any `XML` document can be transformed into a set of facts and loaded into the rule engine memory.

- *Step 2 - Converting Generic Facts into User-Defined Data Structure*: Once, the data is loaded in the memory, the facts could be used directly, however it is possible that all the data is not relevant for a particular query, or that the data must be reformatted prior to use with external decision rules. The *Facts transformer* converts the generic facts into enhanced facts. This conversion is achieved by introducing a set of *transformation rules*, whose goal is to transform these generic facts into a meaningful format. Although this step could be optional, it allows formatting of the raw data into structured data, to enable the writing of reasoning rules.

  These transformation rules are expressed in the language used by the rule engine. In our case, these rules are encoded using the CLIPS syntax. It allows a high level of flexibility in the transformation process and in the definition of the final structure of the data. Each time that a new schema is encountered, end users only have to create or update some of the transformation rules to create schema-specific enhanced facts. Transformation rules can also be used to trigger additional rules when a given condition is detected.
- *Step 3 - Facts Processing*: the *Facts Processor* modifies the enhanced facts with a set of *decision rules*.

Example The example shown in Figure 5 presents the processing of a fragment of a `Web Service Level Agreement` (WSLA) document (1). This part of a WSLA describes the parties involved in a contract: two companies, the provider "`ProviderA`" and the client "`Xinc`". The equivalent document once loaded as generic facts is shown as (2). It is possible to recreate the `XML` document from this set of facts.

One transformation rule is shown as (3). This rule creates a new "enhanced" fact representing the contract between these two companies. It also shows that the newly created fact (4) can be extended with additional information that was not within the initial document. In this example, we have assumed that each contract is identified by an internal unique identifier, so when a new contract is loaded into the rule engine memory, the transformation rule generates a new identifier and associates it with the fact representing the contract. The rule also calls the `RecursiveRemove` function, one of the helper functions provided with the analysis tool. This function recursively removes the fact passed as parameter and its children.

A decision rule (5) is launched, this rule checks if a contract exists between the company "`ProviderA`" as provider and the company "`Xinc`" as client. Finally, the result (6) of the computation, here a Boolean value, is returned.

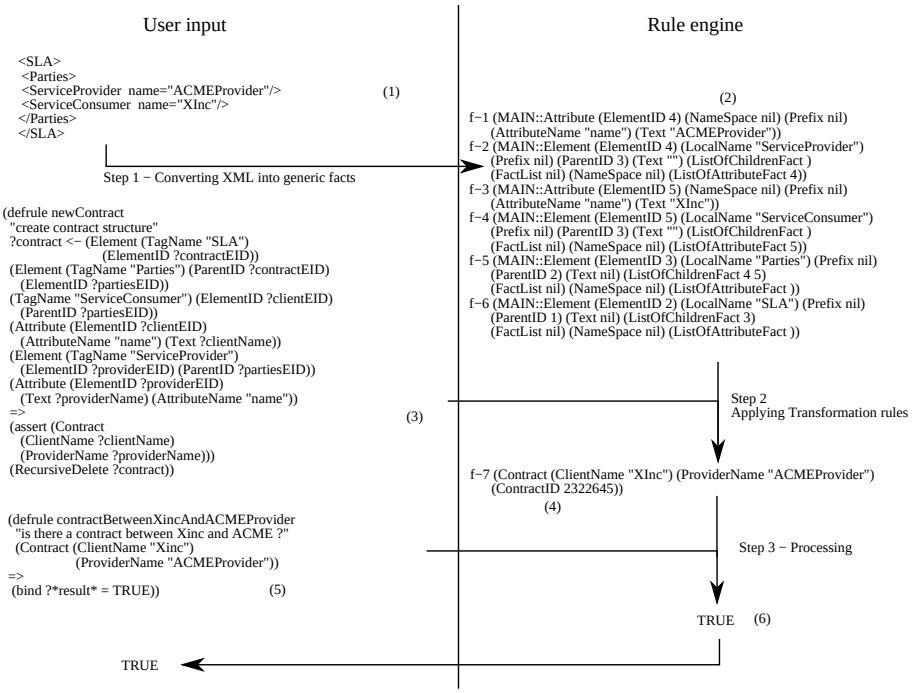

Fig. 5. Example

5 Evaluation and Results

5.1 Bioclimatic Modelling

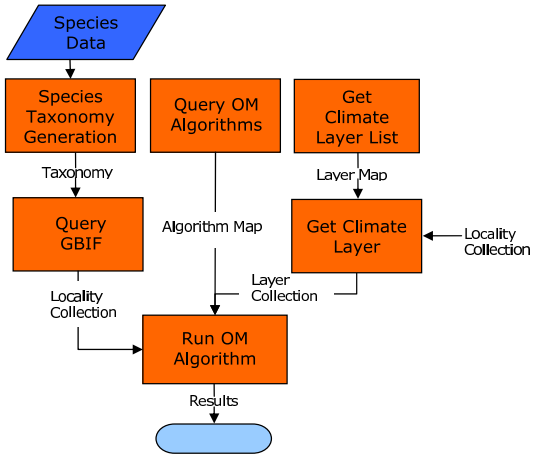

Fig. 6. Bioclimatic Modelling in BDW

Our workflow representing the bioclimatic modelling of species distribution is from the BioDiversityWorld(BDW) project [19]. In Figure 6, given a set of locality data for a species, a climate preference profile is produced by referring to present day climate data to produce a 'climate envelope'. This is then used with a specific selected Open Modeller (OM) algorithm by interpolating the climatic data at the points of locality of specimens producing a bioclimatic model. Use of bioclimatic modelling will allow the prediction of how species will be distributed under changing climate. Such distributions projected upon a world map allows a scientist (or policy maker) to examine where a conservation priority area should be in the future for that species. It is possible to have various case that leads to the evaluation of the projected world map produced by the given workflow [15].

5.2 Evaluation

We present an evaluation of our trust framework using the decision tree in the context of the BDW workflow scenario. Three sets of *p-assertions* that describes an enacted workflow consisting of two services of the BDW scenario; (1) run OM Algorithm as a GUI service that has the algorithm and the species locality data; and (2) get Climate Layer service gets the climate data. One of the set was produced with services functioning correctly. Out of the other two sets, for one set (*p-assertions*) were produced with a dummy Get Climate Layer service that interpreted data incorrectly. The last set was produced by services whose

p-assertions were inadequate (i.e. not providing enough data to answer user queries). These two sets led to some conflicts with existing *p-assertions*, generating scenarios for evaluation. The queried three sets of p-assertions data are passed individually though the decision tree nodes for analysis (figure 8 provides a part of the decision tree for illustration). Trust values are calculated using the algorithm of eq.1 at different nodes for the paths followed in the decision tree (see figure 7). Explaining each analysis node implemented is beyond the scope of this paper. The following describes and illustrates some analysis performed and the results of processing the three sets of *p-assertions* data that generated three paths plotted in figure 7.

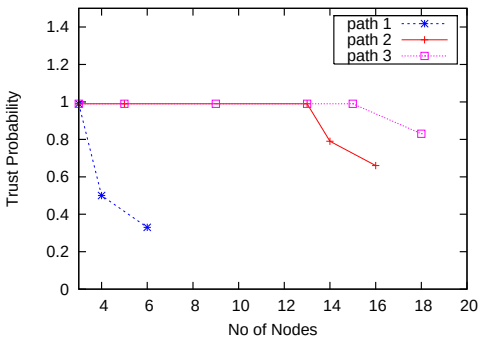

Fig. 7. Trust values for 3 path scenario

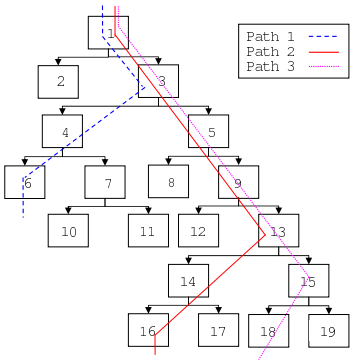

Fig. 8. Decision Tree

Path 1: Traversing the first set of *p-assertions* reaches node 6 (figure 7). This indicates that the analysis performed to check the presence of adequate interaction information is not enough to identify the workflow that was enacted. For example, the negative result at node 4 reveals the lack of relevant relationship information between events that occurred during the workflow enactment. The trust value calculated at this point is 0.33.

Path 2: Traversing the second set of *p-assertions* reaches node 16 (path 2 in figure 7). Until node 13 it indicates that all the necessary information is present in this set of p-assertions to recreate the physical workflow, trust values are at their highest. At this node conflicts in the data passed during an interaction between a sender and a receiver is detected, thus a negative path to node 14 is taken. At node 14 the evaluator inputs his requirement for the parameter "precipitation". This parameter is not detected in the p-assertions, thus the analysis ends at node 16 giving 0.66 as the trust value for the result. Effect of these two analysis can be seen in figure 7. Line of path 2 is pictured falling towards the right-hand – showing how trust value reduces with increased negative outcomes.

Path 3: The third analysis on*p-assertions* ends at node 18, giving a trust value of 0.83 (path 3 in figure 7). Although no conflict in the data was found (at node 13), the parameter requirement (same as in the previous case) at node 15 is not

detected in this *p-assertions* set. This results in an end of the analysis at node 18.

In path 3 the trust value of 0.83 signifies that it is reduced due to the requirement specified by the user. Thus, another user evaluating the same provenance of the workflow without any parameter requirements would have a trust value of 0.99 (node 19 in figure 8). It can therefore be concluded that the trust model indicates that a value of 0.99 produced by following the decision tree indicates a complete trust within the result. Any value below this indicates a possible abnormality in the workflow under evaluation.

6 Conclusion

This paper demonstrates that provenance information captured from a workflow enactment engine could be used for determining the "trustworthiness" of the result generated from such enactment. An analysis tool based on the JESS rule engine has been presented, that may be used to perform subsequent analysis on such provenance information—aiming to automatically calculate trust measures for workflow result. Probabilistic trust measures are calculated using a decision process making use of JESS rules, implemented within an analysis tool. Using this approach, a autonomic workflow enactment engine is able to chose the most trustworthy service. The analysis tool is used to automate the trust assessment process.

References

1. Groth, P., Jiang, S., Miles, S., Munroe, S., Tan, V., Tsasakou, S., Moreau, L.: An architecture for provenance systems. Technical Report, Electronics and Computer Science, University of Southampton (2006)
2. BioDiversityWorld (BDW) Project, http://www.bdworld.org (2005)
3. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: Proceedings of the Twelfth International World Wide Web Conference. (2003)
4. Sabater, J., C.Sierra: Regret: a reputation model for gregarious societies. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agents Systems. (2002)
5. Maximilien, E.M., Singh, M.P.: Toward autonomic web services trust and selection. In: Proceedings of 2nd International Conference on Service Oriented Computing (ICSOC 2004). (2004)
6. Yu, B., Singh, M.P.: A social mechanism of reputation management in electronic communities. In: Proceedings of the Second International Conference on Trust Management(iTrust'04). (2004)
7. Witkowski, M., Aritikis, A., Pitt, J.: Experiments in building experiential trust in a society of objective-trust based agents. Trust in Cyber-societies (2001) 111–132
8. Abdul-Rahman, A., Hailes, S.: Using recommendations for managing trust in distributed systems. In: Proceedings IEEE Malaysia International Conference on Communication. (1997)

9. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Stanford Digital Library Technologies Project (1998)
10. Falcone, R., Castelfranch, C.: Social trust: A cognitive approach. Trust and Deception in Virtual Societies Journal (2001) 55–90
11. Shaikh Ali, A., Rana, O.F., Al-Ali, R.J.: Evidence-aware Trust Model for Dynamic Services. In: High Performance Computing: Paradigms and Infrastructure. (2005)
12. Yang, S.J.H., Hsieh, J.S.F., Lan, B.C.W., Chung, J.Y.: Composition and evaluation of trustworthy web services. In: BSN '05: Proceedings of the IEEE EEE05 international workshop on Business services networks, Piscataway, NJ, USA, IEEE Press (2005) 5–5
13. Liu, W.: Trustworthy service selection and composition - reducing the entropy of service-oriented web. In: 3rd IEEE International Conference on Industrial Informatics (INDIN). (2005)
14. Milanovic, N., Malek, M.: Architectural support for automatic service composition. In: IEEE International Conference on Services Computing (SCC05). (2005)
15. Rajbhandari, S., Wootten, I., Rana, O.: Evaluating provenance-based trust for scientific workflows. In: Sixth IEEE International Symposium on Cluster Compution and the Grid(CCGrid06), Singapore (2006) 365–372
16. Bernardo, J.M., Smith, A.F. In: Bayes Theorem. John Wiley & Sons, West Sussex, England (May 2000) 116–117
17. C. L. Forgy: Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. Artificial Intelligence **19** (1982) 17–37
18. Grid Provenance Project, http://gridprovenance.org (2005)
19. Jones, A., White, R., Pittas, N., Gray, W., Sutton, T., Xu, X., Bromley, O., Caithness, N., Bisby, F., Fiddian, N., Scoble, M., Culham, A., P.Williams: BiodiversityWorld: An architecture for an extensible virtual laboratory for analysing biodiversity patterns. In: UK e-Science All Hands Meeting, EPSRC, Nottingham, UK (2003) 759–765