17. An Ontology for Static Knowledge Provenance

Mark S. Fox and Jingwei Huang
Enterprise Integration Laboratory, University of Toronto
{msf, jingwei}@eil.utoronto.ca

Knowledge Provenance (KP) is proposed to address the problem about how to determine the validity and origin of information/knowledge on the web by means of modeling and maintaining information sources and dependencies as well as trust structures. Four levels of KP are introduced: Static, Dynamic, Uncertain, and Judgmental. In order to give a formal and explicit specification for the fundamental concepts of KP, a static KP ontology is defined in this paper.

1. INTRODUCTION

With the widespread use of Internet and telecommunication technologies that make information globally accessible, knowledge/information validity becomes a crucial factor for enterprise integration, as well as knowledge management within or across enterprises. The validation of parts catalogue information, product requirements, financial information, etc. can be quite costly. For example, an aerospace company designed a device without knowing the NASA approved parts catalogue it was using had been replaced by a newer version, thereby forcing a redesign, delay in delivery and cost overrun.

Knowledge Provenance (hereafter, referred as KP) has been proposed to create an approach to determining the origin and validity of web information by means of modeling and maintaining information sources and dependencies, as well as trust structures. The major questions KP attempts to answer include: Can this information be believed to be true? Who created it? Can its creator be trusted? What does it depend on? Can the information it depends on be believed to be true? This proposed approach could be used to help people and web software agents to determine the validity of web information.

Philosophically, we believe the web will always be a morass of uncertain and incomplete information. But we also believe that it is possible to annotate web content to create islands of certainty. Towards this end, we introduce 4 levels of provenance that range from strong provenance (corresponding to high certainty) to weak provenance (corresponding to high uncertainty). Level 1 (static KP (Fox&Huang2003)) develops the fundamental concepts for KP, and focuses on provenance of static and certain information; Level 2 (Dynamic KP (Huang&Fox2003B)) considers how the validity of information may change over time; Level 3 (Uncertainty-oriented KP (Huang&Fox2004)) considers uncertain truth value and uncertain trust relationships; Level 4 (Judgment-based KP) focuses on social processes necessary to support provenance. Since static KP is the foundation to develop other levels of KP, an explicit formal description is expected.

This paper defines static KP ontology in First Order Logic. Following the ontology development methodology of Gruninger & Fox (1995), we specify static KP ontology in 4 steps: (i) provide a motivating scenario; (ii) define informal competency questions for which the ontology must be able to derive answers; (iii) define the terminology (i.e., predicates); (iv) define the axioms (i.e., semantics).

This paper is organized as follows: Section 2 introduces related research. Section 3,4,5 and 6 define a static KP ontology in 4 steps as stated above. Section 7 introduces implementation. Section 8 gives a summary and a view on future work.

2. RELATED RESEARCH

Interest in addressing the issue of web information trustworthiness has appeared under the umbrella of the "Web of Trust" that is identified as the top layer of The Semantic Web (see (Berners-Lee 2003) slide 26, 27). Digital signature and digital certification ((Simon *et al*, 2001)) play important roles in "Web of Trust". However, they only provide an approach to certify an individual's identification and information integrity, and they do not determine whether this individual could be trusted. Trustworthiness of the individual is supposed to be evaluated by each application. For the purpose of secure web access control, Blaze *et al*, (1996) first introduced "decentralized trust management" to separate trust management from applications. Since then, trust management has grown from web access control to more general trust concerns in various web applications. Although tightly related, in the context of knowledge provenance, trust management only considers direct and indirect trust relationships to information creators but does not consider the dependencies among information units. KP addresses both trust relationships and the dependencies among information units. In addition, coming from an automated reasoning perspective, "Inference Web (IW)" (McGuinness&Silva2003) enables information creators to register proofs with provenance information in IW, and then IW is able to explain the provenance of a piece of requested knowledge. IW provides provenance information (registered by creators) for users to support them deciding by themselves to trust or not trust the requested knowledge.

In addition, information source evaluation criteria, such as, Authority, Accuracy, Objectivity, Currency and Coverage, have been developed in library and information science, and have been extended to online information (Alexander,1999).

Finally, the technologies developed in Semantic Web (Berners-Lee,2001) provide an approach to the web implementation of KP. Many technologies developed in AI, such as Truth Maintenance System (de Kleer, etc, 1989), etc., provide a basic approach for knowledge representation and reasoning in KP.

3. MOTIVATING SCENARIO

In the following, the underlying concepts of Static KP are explored in the context of two case studies.

Case 1: Asserted Information

Consider a proposition in a document found on the intranet in an enterprise. This proposition states that "a delay of more than one minute in answering a phone call may cause the customer to be unsatisfied." From a provenance perspective, there are three questions that have to be answered: 1) What is the truth value of this proposition? 2) Who asserted this proposition? 3) Should we believe the person or organization that asserted it? In this example, a further examination of the text of the web document provides the answers: It can be believed as a true proposition, asserted by a retired customer service manager, who most people in the company believe is an authority on the subject. Questions are: (1) what is the basis for us to believe this proposition as true? (2) how can the provenance process be formalized?

Case 2: Dependent Information

Consider the following proposition found in another web document: "This new approach to reduce response-delay to less than one minute may increase customer loyalty, because a delay of more than one minute in answering a phone call may cause the customer to be unsatisfied." This is actually two propositions composed of a premise, " a delay of more than one minute in answering a phone call may cause the customer to be unsatisfied." and a conclusion, " This new approach to reduce response-delay to less than one minute may increase customer loyalty." Just as in the previous case, the same questions need to be answered for each proposition. What makes this case more interesting is that answering these questions is dependent upon propositions found in other web pages. There are two types of dependency occurring. First, the truth of the premise is dependent on the truth of the proposition found in another web document. Second, the truth of the conclusion depends on the truth of the premise and upon some hidden reasoning that led to the deduction. These types of propositions are called "dependent propositions" in KP.

It is common to find information in one document that is reproduced in another. The reproduction of a proposition in a second document leads to an equivalence relationship between the two propositions, i.e., the truth values of the two propositions are equivalent. However, the relationship is also asymmetric; one proposition is a copy of the other. The copy of one proposition is classified as "equivalent information". Furthermore, a proposition can be derived using logical deduction. Hence, the truth value of the derived proposition depends on the truth values of its antecedent propositions. This type of derived proposition is classified as "derived information".

Returning to the example, determining the provenance of the premise requires that we link, in some way, the premise to the proposition in the other web document from which it is copied. The same is true of the conclusion. Minimally, we should link it to its premise, maximally we should link it to the axioms that justify its derivation. These links will also require some type of certification so that we know who created it and whether it is to be trusted.

From these two cases, a number of concepts required for reasoning about provenance emerge:

- Text is divided into propositions. Once so designated, they are assumed to be indivisible.
- An proposition must have a digital signature.
- An assertion is believed to be true, if the information user trusts the person or organization that created the assertion in the corresponding topic.
- As propositions are reused across the web, a link between where it is used and where it came from must be maintained. These links, or dependencies, must be included in the digital signatures with propositions.

- Dependencies can be simple copies, or can be the result of a reasoning process. If the latter, then axioms used in the reasoning should also be identified and signed by an acceptable organization.

4. INFORMAL COMPETENCY QUESTIONS

What static KP needs to answer, called informal competency questions, are identified as follows. These questions define the requirements to Static KP.

- Is this proposition true, false, or unknown?
- Who created this proposition?
- What is the digital signature verification status?
- Which knowledge fields does this proposition belong to?
- In these fields, can the information creator be trusted?
- Does the truth of this proposition depend on any other propositions? If so, which ones?

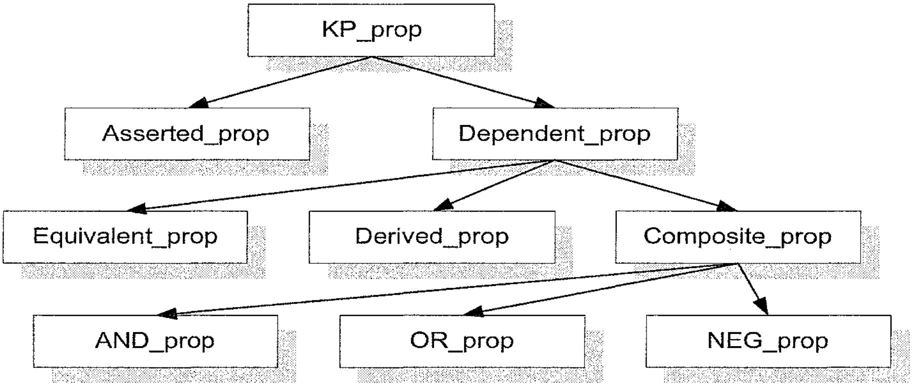

Figure 1. Proposition Taxonomy in Knowledge Provenance

5. TERMINOLOGY

There are five main classes in the static KP ontology: Propositions, Documents, Information Sources, Trust Relationships and Signature Status.

Propositions

The basic information unit in KP is a proposition. KP-Prop is the most general concept used to represent propositions in a document. From our motivating scenario in section 3 and the natures of propositions, we prefer to depict the taxonomy of propositions in KP as shown in Figure 1. An Asserted_prop is an assertion that is not dependent on any other propositions; a Dependent_prop is a proposition which truth is dependent on other propositions; an Equivalent_prop is a quotation that is a copy and its truth value is the same as the proposition it depends on; a Derived_prop is a derived conclusion based on some premises; a Composite_prop could be the "and"/ "or" / "negation" of other proposition(s).

Table I defines the predicates for depicting a KP proposition and its attributes.

Table I. Predicates depicting a KP proposition and its attributes

Predicate	Description
type(x, "KP_prop")	x is defined to be a proposition, signified by being of type KP_prop.
proposition_content(x,s)	s is the content of the proposition x. In html files, the content of a proposition usually is a string; in xml files, the content of a proposition can be an xml element.
assigned_truth_value (x,v)	Proposition x has a truth value v assigned by proposition creator.
trusted_truth_value(a,x,v)	Agent a trusts that proposition x has a truth value v. v may be one of "True", "False", or "Unknown".
type(x, "asserted_prop")	x is an assertion and does not depend upon any other proposition.
type(x, "dependent_prop")	x is a proposition whose truth value is dependent upon another proposition. Dependent-prop class is further divided into 3 subclasses: equivalent-prop, derived-prop, and composite-prop.
type(x, "equivalent_prop")	An equivalent-prop is a copy of and its truth value is the same as the proposition it depends on.
type(x, "composite_prop")	Composite-prop is defined to be the logical combination of its constituent propositions. A composite-prop is divided into 3 subclasses: neg-prop, and-prop, and or-prop.
type(x, "derived_prop")	A derived-prop indicates that the proposition is a derived conclusion based on some premises. For example, derived-prop B has dependency-link pointing to composite-prop A, meaning that A is a premise of B.
is_dependent_on(x, y)	Proposition x is dependent on proposition y. x is called dependent proposition, and y is called support proposition.
has_same_content(x,y)	Proposition x has the same proposition content as y.

Documents

To facilitate the determination of the provenance of a proposition, properties of the document in which it appears may need to be considered. For example, knowing who created the document may be important in determining the validity of a proposition within. A document can be any type of file. For the purposes of this paper, we restrict our attention to standard web files such as: html files, xml files, and xhtml files. Following are document related KP predicates:

Predicate	Definition
type(x, "document")	*x* is defined to be a KP document.
in_document(y,d)	Proposition *y* is contained in document *d*.

Information Source and Signature

For any document and proposition its creator can be defined. Along with it can be defined a digital signature and the verification status of the signature. Assume that a digital signature validation software provides the result of signature verification.

Predicate	Description
has_infoCreator(x,c)	KP-prop or Document *x* has infoCreator *c*. Here, infoCreator may be either creator or publisher.
has_signature(x, s)	The proposition or document x has a signature s.
has_sig_status(x, v)	The digital signature verification status of *x* is *v*, where *v* may be one of three status: "Verified"-- the signature is verified successfully; "Failed"-- the signature verification is failed; and "NoSignature"-- do not have digital signature.

Trust Relationships

In section 3 we stated that KP is context sensitive, where the context is largely associated with trust relationships that define the provenance requester trusts whose propositions in what topics. A trust relationship in KP is defined as a of triple *(a, c, f)* where the provenance requester (information receiver) *a* "trusts" information creator *c* in a topic or a specific knowledge field *f*, here, "trust" means that *a* believes any proposition created by *c* in field *f* to be true. (Note: The mathematical definition of a trust relation should be a set of triples $\{(a, c, f)\}$. A triple *(a, c, f)* is called a trust relationship in this paper). The following defines the trust related predicates:

Predicate	Description
trusted_in(a, c, f)	Provenance requester *a* trusts information creator *c* in knowledge field *f*.
trusted(x, a)	Proposition *x* is trusted by agent *a*. That means its information creator is trusted by *a* in one of the fields which proposition *x* belongs to.
in_field(x,f)	Proposition *x* belongs to knowledge field *f*.
subfieldOf(x,y)	Knowledge field *x* is a sub-field of knowledge field *y*

6. AXIOMS

In the following, a set of axioms is defined to specify truth conditions of KP-props. Basically, the truth value of an asserted proposition depends on if the proposition is "trusted"; the truth value of an equivalent proposition depends on the truth value of the proposition that this equivalent proposition points to by its dependency-link; the truth value of a derived proposition depends on if the proposition is "trusted" and if its support KP-prop is true. In addition, a KP-prop is "trusted", if the creator or publisher of the proposition is trusted in one of the fields of the proposition, and the digital signature verification status is "Verified". Finally, note that the "close world assumption" is applied to handle "not" in this paper.

Asserted Propositions

An asserted-prop is trusted to have its truth value as assigned, if the asserted-prop is trusted by the provenance requester.

Axiom SKP-1:

for-all (a,x,v)
((type(x, "asserted_prop") ^ trusted(x, a) ^ assigned_truth_value(x, v))
→ *trusted_truth_value(a, x, v)).*

A KP-prop is "trusted", if the creator or publisher of the proposition is "trusted" in one of the fields of the proposition, and the digital signature verification status is "Verified".

Axiom SKP-2:

for-all (a,x,f,c,w)
((type(x, "KP-prop") ^ has_sig_status(x, "Verified") ^ has_infoCreator(x, c)
^ in_field(x, f) ^ trusted_in(a, c, w) ^ subfield_of(f, w))
→ *trusted(x, a)).*

For a KP-prop that has no creator specified, the creator of the document is the default creator of the KP-prop.

Axiom SKP-3:

for-all (x, d, c)((type(x, "KP-prop")
^ (not(exist (c2) has_creator(x, c2)))
^ in_document(x, d) ^ has_creator(d, c))
→ *has_creator(x, c)).*

If a proposition does not have a creator, then the digital signature verification status of the KP-prop is determined by the digital signature verification status of the document.

Axiom SKP-4:

for-all (x, d, c, v)((type(x, "KP-prop")
^ (not (exist (c2) has_creator(x, c2)))
^ in_document(x, d) ^ has_creator(d, c) ^ has_sig_status(d, v))
→ *has_sig_status(x, v)).*

Equivalent Propositions

The trusted truth value of an equivalent-prop is the same as the trusted truth value of its support proposition, if this equivalent-prop exactly has the same proposition-content as its support proposition has.

Axiom SKP-5:

for-all (a, x, y, v) ((type(x, "equivalent_prop")
^ is_dependent_on(x, y) ^has_same_content(x,y)
^ trusted_truth_value(a, y, v))
→ *trusted_truth_value(a, x, v)).*

Composite Propositions

The trusted truth value of a neg-prop is the negation of the trusted truth value of the KP-prop it is dependent on.

Axiom SKP-6:

for-all (a, x, y)((type(x, "neg_prop")
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, "True"))
→ trusted_truth_value(a, x, "False")).

Axiom SKP-7:

for-all (a, x, y)((type(x, "neg_prop")
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, "False"))
→ trusted_truth_value(a, x, "True")).

The trusted truth value of an and-prop is "True" if all its support KP-props are "True"; and the trusted truth value of an and-prop is "False" if at least one of its support KP-props is "False".

Axiom SKP-8:

for-all(a, x)((type(x, "and_prop")
^for-all (y) (is_dependent_on(x, y) -> trusted_truth_value(a, y, "True")))
→ trusted_truth_value(a, x, "True")).

Axiom SKP-9:

for-all(a, x)((type(x, "and_prop")
^(exist(y) (is_dependent_on(x, y) ^ trusted_truth_value(a, n, "False"))))
→ trusted_truth_value(a, x, "False")).

The trusted truth value of an or-prop is "True" if at least one of its support KP-props is "True"; and the trusted truth value of an or-prop is "False" if all its support KP-props are "False".

Axiom SKP-10:

for-all(a, x)((type(x, "or_prop")
^ (exist (y) (is_dependent_on(x, y) ^ trusted_truth_value(a, y, "True"))))
→ trusted_truth_value(a, x, "True")).

Axiom SKP-11:

for-all(a, x)((type(x, "or_prop")
^ (for-all (y) (is_dependent_on(x, y) ^ trusted_truth_value(a, y, "False"))))
→ trusted_truth_value(a, x, "False")).

Derived Propositions

The trusted truth value of a derived proposition is "True" or "False" as specified, if it is "trusted" and its support KP-prop (condition) is "True". Note that the axioms used to derive the truth value do not have to be included as part of the dependency.

Axiom SKP-12:

for-all (a, x, y, v)((type(x, "derived_prop")
^ trusted(x, a) ^ assigned_truth_value(x, v)
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, "True"))
→ trusted_truth_value(a, x, v)).

Default assigned_truth value

The default truth value of an asserted or derived proposition assigned by the proposition creator is "True".

Axiom SKP-13:

for-all (a, x, y, v)((type(x, "Asserted_prop")^ type(x, "Derived_prop")
^ triple(x, assigned_ truth_value, v))
→ assigned_truth_value(a, x, v)).
for-all (a, x, y, v)((type(x, "Asserted_prop")^ type(x, "Derived_prop")
^ not (triple(x, assigned_ truth_value, v)))
→ assigned_truth_value(a, x, "True")).

Default trusted_truth value

The default trusted truth value of a proposition is "Unknown".

Axiom SKP-14:

for-all (a, x, v)((type(x, "KP_prop")
^ not (trusted_truth_value(a, x, "True"))
^ not (trusted_truth_value(a, x, "False")))
→ trusted_truth_value(a, x, "Unkown")).

7. IMPLEMENTATION

To apply KP in practice, information creators need to annotate web documents with KP metadata, users (provenance requesters) need to define their trust relationships, and a KP reasoner conducts provenance reasoning on annotated web documents.

In order to facilitate the annotation of web documents with KP metadata and define trust relationships, we have defined a KP markup language in RDFS (Resource Description Framework Schema) (Brickley&Guha, 2004). The following is a piece of example containing only one proposition in a web document annotated with kp metadata. An entire annotation example can be found in (Fox&Huang2003).

```
<kp:Derived_prop rdf:id="ReduceDelay"
    is_dependent_on="#ProblemOfDelay"
    creator ="Tim Levy"
    in_field ="Custom Relationship Management">
            The new approach to reduce response-delay to
            less than one minute may increase customer loyalty.
</kp:Derived_prop>
```

We have implemented the KP reasoner with Prolog. The system can infer the truth of any KP-prop. In the above example, assume that information user *A*, who requests the provenance of this proposition, trusts Tim Levy in field "Custom Relationship Management", and the digital signature verification status of the proposition is "verified", to determine the trusted truth value of this derived proposition, the system applies axiom SKP-12, and then the main goal *trusted_truth_value(A, "ReduceDelay",v)* is divided into several sub-goals: *trusted("ReduceDelay",A)* is solved by applying axiom SKP-2; *assigned_truth_value("ReduceDelay",v)* is solved by applying axiom SKP-13, and *v* is bound as *"True"*; from kp metadata, *is_dependent_on("ReduceDelay", "ProblemOfDelay")* is true; this leads to solve sub-goal *trusted_truth_value(A, "ProblemOfDelay", "True"))* by applying axiom

SKP-1. The process to solve this subgoal is similar. If this last sub-goal is solved, then the main goal is solved and the returned trusted truth value *v* is "True"; otherwise, the main goal is failed, the system returns trusted truth value of "*Unknown*" by applying axiom SKP-14.

8. DISCUSSION AND FURTHER WORK

Knowledge Provenance is an approach to determining the validity and origin of information/knowledge by means of modelling and maintaining information source and dependencies, as well as trust structures. Four levels of KP are introduced: Static, Dynamic, Uncertain, and Judgmental. In order to give a formal and explicit specification for the fundamental concepts of KP, a static KP ontology was defined in this paper. A KP markup language was designed with RDFS; a KP reasoner that traces the web documents annotated with kp metadata and deduces the origin and validity of requested information was implemented in Prolog.

Based on this formal static KP model, we have developed a dynamic KP model (Huang&Fox2003) that determines the validity of information in a world where the validity of a proposition and trust relationships are changing over time. Furthermore, an uncertainty-oriented KP model that introduces "trust degree" to represent uncertain trust relationships and "certainty degree" to represent uncertain truth value has been developed (Huang&Fox2004).

We will continue our work towards judgment-based KP to develop a formal "social" process representing trust propagation in social networks. In addition, a web based KP reasoner will be implemented to deduce the origin and validity of requested web information by tracing web documents across the web.

As stated earlier in the paper: "we believe the web will always be a morass of uncertain and incomplete information". The challenge therefore is to create models and processes that will enable the validation of as much information as possible.

This research was supported, in part, by Bell University Laboratory and Novator Systems, Ltd.

REFERENCES

Alexander, J. E., and Tate, M.A., (1999), Web Wisdom: how to evaluate and create information quality on the web, Lawrence Erlbaum Associates Publishers.

Berners-Lee, T., (1998), Semamtic Web Road Map, http://www.w3.org/DesignIssues/Semantic.html

Berners-Lee, T., Hendler, J., and Lassila, O., (2001), "The Semantic Web", Scientific American, May 2001.

Berners-Lee, T., (2003), The Semantic Web and Challenges,. http://www.w3.org/2003/Talks/01-sweb-tbl/

Blaze, M., Feigenbaum, J. and Lacy, J., (1996), Decentralized Trust Management, Proceedings of IEEE Conference on Security and Privacy, May, 1996.

Brickley, D. and Guha, R.V., RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

de Kleer, J., Forbus, K., McAllester,D., Truth Maintenance Systems (Tutorial SA5), Int. Joint Conference on Artificial Intelligence, SA5-182~225, 1989.

Fox, M. S., and Huang, J., (2003), "Knowledge Provenance: An Approach to Modeling and Maintaining the Evolution and Validity of Knowledge", EIL Technical Report, University of Toronto. http://www.eil.utoronto.ca/km/papers/fox-kp1.pdf

Gruninger, M., and Fox, M.S., (1995), "Methodology for the Design and Evaluation of Ontologies", Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal.

Huang, J. and Fox, M. S., (2003B), " Dynamic Knowledge Provenance ", EIL Technical Report, University of Toronto. http://www.eil.utoronto.ca/km/papers/kp2-TR03.pdf

Huang, J., and Fox, M.S., (2004), "Uncertainty in Knowledge Provenance", in Bussler, C. Davies J., Fensel D., Studer R. (eds.) The Semantic Web:Research and Applications, Lecture Notes in Computer Science 3053, Springer, 2004, PP.372-387.

Huhns, M.H., Buell, D. A., (2002), Trusted Autonomy, IEEE Internet Computing, May. June 2002.

Khare, R., and Rifkin, A., (1997), "Weaving and Web of Trust", World Wide Web Journal, V.2, pp.77-112.

Simon, E., Madsen, P., Adams, C., (2001), An Introduction to XML Digital Signatures, Aug., 2001. http://www.xml.com/pub/a/2001/08/08/xmldsig.html