Huang, J., and Fox, M.S., (2004). "Dynamic Knowledge Provenance", Proceedings of Business Agents and Semantic Web Workshop, pp. 372-387, National Research Council of Canada, London Canada.

Dynamic Knowledge Provenance

Jingwei Huang Mark S. Fox
Enterprise Integration Laboratory, University of Toronto
40 St. George Street, Toronto, ON M5S 3G8 Canada
{ jingwei, msf}@eil.utoronto.ca

ABSTRACT

Knowledge Provenance (KP) is proposed to address the problem of how to determine the validity and origin of web information by introducing standards and methods for modeling and maintaining the evolution and validity of information/knowledge on the web. Four levels of KP including Static KP, Dynamic KP, Uncertain KP, and Judgmental KP are proposed. This paper focuses on Dynamic KP to address the problem of how to determine the validity of web information over time. A Dynamic Knowledge Provenance ontology is defined, and an example is given to illustrate how to annotate web document with dynamic KP metadata and how to use dynamic KP axioms to infer the validity of the web information.

Keywords

Provenance, Trust, Information dependency, Semantic Web.

1. INTRODUCTION

The emerging web technologies of the Semantic Web, Web Services, and Semantic Web Services will technically enable agents communicate with agents on the Web. However, how to determine the information trustworthiness in the communication among agents (including both people and software) still remains an open issue. “Web of trust”, as the top layer of the Semantic Web, is an important area to be explored ([Berners-Lee, 2003] slide 26&27). No doubt, it is crucial for both people and software agents to determine information trustworthiness before they make a decision especially on the Web with dynamic information sources and dynamic business partnerships.

Provenance (KP) is proposed by Fox and Huang (2003) to address the problem of how to determine the validity and origin of web information. The problem arises from many directions: information may no longer be relevant (e.g., discontinued products or old operating procedures), may contain incorrect information (e.g., news stories), and may even be outright lies. For example, in 1999, two men posted fraudulent corporate information on electronic bulletin boards, which caused the stock price of a company (NEI) to soar from $0.13 to $15, resulting in their making a profit of more than $350,000 [14]. This example reveals a problem: anyone can publish information on the web, however, the information may be true or false, current or outdated. Therefore, the means to discern the differences is expected. Knowledge provenance aims at creating an approach for both people and software agents to determine the origin and validity of web knowledge by means of modeling and maintaining information sources and dependency, as well as trust structure.

Philosophically, we believe the web will always be a morass of uncertain and incomplete information, and the majority of it generated manually. But we also believe that it is possible to annotate web content to create islands of certainty[1]. Towards this end, four levels of KP have been identified. Level 1 (Static KP) focuses on provenance of static and certain information; Level 2 (Dynamic KP) considers how the validity of information may change over time; Level 3 (Uncertain KP) considers information whose validity is inherently uncertain; Level 4 (Judgment-based KP) focuses on social processes necessary to support provenance. Static KP has been studied in [7] and uncertain KP has been studied in [11].

This paper focuses on Dynamic KP. In the real world, the validity of information may change over time. Consider the supply chain, the prices of products change over time, inventory changes over time, warehouse space changes over time, etc. For example, a computer retailer receives a proposition from its CPU supplier stating “150 Intel P4 Processors at 3.06GHZ available” (valid from 2003-04-14 to 2003-05-16), which is true in the specified period, but may be false before or after the period. This paper introduces Dynamic Knowledge Provenance to address the problem of how to determine the validity of information that changes over time.

The content of this paper is organized as follows: Section 2 discusses related research; Section 3 reviews the basic concepts of Static KP; Section 4 provides motivating scenarios for Dynamic Knowledge Provenance. Section 5 presents Dynamic KP ontology. Section 6 introduces the implementation and example. Finally, we provide a summary and a view on future work in section 7.

2. RELATED RESEARCH

Interest in addressing the issue of web information trustworthiness has appeared under the umbrella of the "Web of Trust".

No doubt, digital signature and digital certification ([3][19][20]) will play important roles in "Web of Trust". The inclusion of an expiration date for a public key provides an important component for Dynamic KP. However, they only provide an approach to certify an individual’s identification and information integrity, but they do not determine whether this individual could be trusted. In the context of knowledge provenance, they can only be used to determine who is the information creator and whether the information is the same as originally defined, but they do not indicate whether the information is trustworthy. Trust related decisions are processed by applications.

Blaze, Feigenbaum and Lacy [5] propose “Decentralized Trust Management” that is a trust management framework separating trust management from applications for the purpose of secure web applications. PolicyMaker [5] is one of representative trust

[1] This perspective differs from one in which knowledge generation is automated and provenance is reduced to recording the proof.

management systems. It accepts the local policies of service provider and the credentials of service requesters as input, and outputs "yes/no" answer or restrictions to the action requested. Trust management provides fundamental concepts for KP. However, in the context of knowledge provenance, it only considers trust relationships but does not consider the dependency among information.

Recently, several projects focusing on trust evaluation using social networks have emerged. Golbeck et al. [9] propose trust networks that extend the FOAF model [6] by introducing levels of trust to describe acquaintance relations. Richardson et al. [18] propose a model of trust management for the semantic web. Both of them focus on transitive trust relationships. However, similar to trust management, they also do not consider dependency among information. Gil and Ratnakar [8] develop an online system called "TRELLIS" that enables users to rate online information, and combine individual views into an overall rating. It could be an effective approach for web information evaluation in web services. However, all of above research does not consider that trust relations may change over time. Dynamic KP model will address this issue.

Coming from an automated reasoning perspective, McGuinness and Silva [15] are developing "Inference Web (IW)" which 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 [2]. Oliver etc. [16] collected hundreds of evaluation criteria from different sources, and consolidate into 125 indicators in 11 groups of criteria. These criteria have been proposed to evaluate web information, and will form one of the basis of what we call Level 4 Judgment-based KP.

Finally, many technologies developed in AI, such as Truth Maintenance System, temporal logic, uncertainty logic, etc., provide basic approach for knowledge representation and reasoning in knowledge provenance.

3. STATIC KNOWLEDGE PROVENANCE

As Dynamic Knowledge Provenance extends Static Knowledge Provenance, this section gives a brief introduction to the basic concepts of Static KP. A detailed description on Static KP can be found in [7].

3.1 Overview of Static KP

The basic unit of web information to be considered in KP is a "proposition". A proposition, as defined in First Order Logic, is a statement that is either true or false. A proposition is the smallest piece of information to which provenance-related attributes may be ascribed.

A proposition may be derived by applying different axioms. Derived propositions may also be dependent upon disjunctions, conjunctions and/or negations of other propositions.

Key concepts underlying Static KP are:

- Text is divided into propositions. Once so designated, they are assumed to be indivisible.
- An asserted proposition must have a digital signature.
- If the truth value of a proposition is to be believed, then the person or organization that signed the proposition must be acceptable to the user of the information.
- 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 also be digitally signed.
- 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 digitally signed by an acceptable organization.

Finally, throughout the above points, the notion of acceptable signing authorities is basic to the analysis of provenance. Consequently, Knowledge Provenance is context sensitive, where the context is defined by a set of signing authorities acceptable to the person requesting provenance.

3.2 Terminology

Propositions

KP-Prop is the most general concept used to represent propositions in a document. The following table defines the predicates for 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 a xml element.
assigned_truth_value [*] ***(x,v)*** :	Proposition *x* has a truth value *v* assigned by proposition creator. *v* may be one of "True" or "False".
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 not dependent 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 original name of this predicate in Static KP is *truth_value*. We change the name into *assigned_truth_value* in this paper.

	the proposition it depends on.
type(x, "composite_prop"):	Composite-prop's is defined to be the logical combination of its constituent propositions. A composite-prop is divided into 3 subclasses: negative-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, which means that A is a premise of B.
is_dependent_on(x, y)	Proposition *x* is dependent on proposition *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 it. 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 and xml 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 digital signature validation software provides the result for 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.
in_document(x,d):	Proposition *x* is contained in document *d*.

Trust Relations

Knowledge Provenance is context sensitive, where the context is defined by a set of signing authorities acceptable to the person requesting provenance information. Provenance is dependent on who the requester trusts. Trust in Knowledge Provenance is defined as a set of triples {(*a, y, z*)} where the information receiver *a* "trusts" information creator *y* in a topic or a specific knowledge field *z*. Here, "trust" means that *x* believes any proposition created by *y* in the field *z* to be true. 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_fields(x,fl):	Proposition *x* is in a field given in field list *fl*.
subfieldOf(x,y):	Knowledge field *x* is a sub-field of knowledge field *y*

3.3 Static Axioms

The following summarizes the axioms for static KP. FOL specification can be found in [7].

- ¬ 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".
- ¬ For an asserted, or derived, or equivalent KP-prop that has no creator specified, the creator of the document is the default creator of the KP-prop.
- ¬ 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.
- ¬ An asserted-prop has its truth value as assigned, if the asserted-prop is trusted by the agent making the provenance request.
- ¬ The trusted truth value of an equivalent-prop is the same as the trusted truth value of the proposition it depends on, if this equivalent-prop is trusted.
- ¬ The truth value of a derived proposition is "True" or "False" as assigned, if it is trusted and its dependency KP-prop (condition) is "True". Note that the axiom used to derive the truth value does not have to be included as part of the dependency.
- ¬ The trusted truth value of a negative-prop is the negation of the trusted truth value of the KP-prop it is dependent on.
- ¬ The truth value of an And-prop is "True" if all its dependency KP-props are "True"; The truth value of an And-prop is "False" if at least one of its dependency KP-props is "False"; and the truth value of an And-prop is "Unknown" if at least one of its dependency KP-props is "Unknown" and none of them is "False".

¬ The truth value of an Or-prop is "True" if at least one of its dependency KP-props is "True"; The truth value of an Or-prop is "False" if all its dependency KP-props are "False"; and the truth value of an Or-prop is "Unknown" if at least one of its dependency KP-props is "Unknown" and none of them is "True".

4. WHAT IS DYNAMIC KNOWLEDGE PROVENANCE?

In the following, the underlying concepts of Dynamic Knowledge Provenance are explored in the context of three case studies.

Consider a story in an IT supply chain composed of a reseller (FS), a distributor (DT) and a manufacturer (HP). The reseller (FS) keeps receiving requests from customers about desktop computers configured with 3.06G Pentium 4 processor, so FS sends a asserted proposition, "There is an increasing market demand for desktops with 3.06G Pentium 4 processor", to its major supplier – distributor (DT). The sales department of the distributor (DT) forwards the message to the product management department which is responsible for product supply. That is, in the terms of KP, the sales department generates an equivalent proposition with the same proposition content as the assertion made by FS. Then, the product management department requests the product information from its major supplier – manufacturer HP. HP replies an asserted proposition, "10,000 desktop PCs configured with 3.06G Pentium 4 processor are available" (effective from 2003-05-26 to 2003-06-01). Based on the asserted proposition made by HP, the equivalent proposition created by the sales department, and some other factors, for example, the distributor is able to order 8000 before 2003-05-31 due to its financial constraints, the product management department recommends to head office a product order plan, (actually a derived proposition,) "We should order 8,000 desktop PCs configured with 3.06G Pentium 4 processor from HP" (effective from 2003-05-26 to 2003-05-31).

Case 1: Asserted proposition's truth value is effective within a specified period

In this example, the truth value of the asserted proposition made by HP is effective during period from 2003-05-26 to 2003-06-01. Assume the distributor trusts HP's product supply information, so the asserted proposition is trusted to be true at any time point within the specified period. But after this period, the truth value of the proposition becomes invalid, so it will no longer be trusted as "true".

Case 2: Information creator is trusted only within a specified period

The Information creator may be trusted only within a specified period also. In the example above, assume there is a contract between the distributor (DT) and the reseller (FS) effective from 2002-04-01 to 2003-12-31. During this period, the distributor trusts the market demand information provided by the reseller to be true. However, if the contract is expired, the distributor will no longer trust the information provided by the reseller anymore. For example, if the contract is effective only from 2002-04-01 to 2003-03-31, the assertion made by the reseller, "There is an increasing market demand for desktops with 3.06G Pentium 4 processor", will not be trusted on 2003-05-26.

Case 3: Derived proposition's truth value is effective within a specified period

Furthermore, the truth value of the derived proposition made by the product management department of DT is effective only during period from 2003-05-26 to 2003-05-31, due to some hidden facts in the derivation. The derived proposition is trusted to be true at a given time point, if the time point is within the specified period of the derived proposition and all its dependency propositions are trusted to be true at the time point.

These examples reveal some important points for building Dynamic Knowledge Provenance.

- The truth value of an asserted/derived proposition may be effective only in a specified period;
- Conjunctive propositional dependencies may give rise to a smaller, or null periods of truth value validity;
- Disjunctive propositional dependencies may give rise to discontinuous periods of truth value validity;
- A Proposition creator may be trusted in a topic only within a specified period. So, trust relations further constrain the periods of truth value validity.

5. DYNAMIC KNOWLEDGE PROVANCE ONTOLOGY

In order to give a formal and explicit specification for Dynamic KP, a Dynamic KP ontology is defined. Following the ontology development methodology of Gruninger & Fox [10], we specify the ontology in 4 steps: (i) provide a motivating scenario that has already been discussed in section 3; (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 section presents informal competency questions, terminology, and axioms.

5.1 Informal Competency Questions

In addition to the informal competency questions that Static KP answers, what Dynamic Knowledge Provenance needs to answer also includes:

- From what time point is the truth value of this asserted proposition effective?
- Until what time point is the truth value of this asserted proposition effective?
- From what time point is the truth value of this derived proposition effective?
- Until what time point is the truth value of this derived proposition effective?
- At a given time point, is the truth value of this proposition effective?
- From what time point is this information creator is trusted in a specified field?
- Until what time point is this information creator is trusted in a specified field?

- At a given time point, is this information creator trusted on a specific topic/field?
- At a given time point, is this proposition trusted?
- At a given time point, what is the trusted truth value of this proposition?

5.2 Terminology

To define dynamic KP axioms for provenance reasoning, the terminology used is defined as follows.

Effective Period

In section 3, we found that the truth value of an asserted or derived proposition may be effective only within a specified period. We call this the proposition's "*effective period*". When the truth value of a proposition is effective at a given time point, the proposition is called "*effective*" at the time point.

We also found that a proposition creator may be trusted only within a specified period. That is, a trust relation element (*a,c,f*), which means provenance agent *a* trusting proposition creator *c* in field *f*, may have effective period also. A trust relation element is called "effective" at a given time point, if the time point is within the effective period of the trust relation element.

In order to describe effective period, we define the following predicates.

Predicate	Description
type(x, "TrustRelationElm")	Object *x* is a trust relation element, i.e., *x* is a triple (*a,c,f*) in a trust relation {(*a,c,f*), … }
effective_from(x, t_1)	*x* is effective from time point t_1. Here, *x* could be Asserted or Derived KP_prop, or trust relation element.
effective_to(x, t_2)	*x* is effective till time point t_2. Here, *x* could be Asserted or Derived KP_prop, or trust relation element.
effective_at(x, t)	KP_prop or trust relation element *x* is effective at time point *t.*

And several predicates defined in Static KP need to be extended with time.

Predicate	Description
trusted_in_during(a,c,f,t_1,t_2)	Agent *a* trusts information creator *c* in knowledge field *f* from time point t_1 to time point t_2. Here, [t_1, t_2] is called trust relation effective period in this paper.
trusted_in(a,c,f,t)	Agent *a* trusts information creator *c* in knowledge field *f* at time point *t,* which is also called trust relation element (*a,c,f*) is effective at time point *t.*
trusted(x,a,t)	Proposition *x* is trusted by agent *a* at time point *t*.

5.3 Axioms

Effective at a time point

At a given time point, if a KP_prop is not effective at the time point, it has trusted truth value of "Unknown". Note:, the "close world assumption" is applied to handle "not" in this paper.

Axiom DKP-1:

for-all (a,x, t, v)
((type(x, "KP_prop") ^ not (effective_at(x,t)))
->trusted_truth_value_at(a,x, ' Unknown' ,t))

The default effective period is "indefinitely", that is, if the effectiveFrom /effectiveTo is not specified, the effectiveFrom / effectiveTo will be negative infinite (denoted as -M) / positive infinite (denoted as +M). In practice, only in the case of that predicate effectiveFrom /effectiveTo is not specified, axiom DKP-2 and 3 will be applied.

Axiom DKP-2:

for-all(x)
(((type(x, ' assertedprop') or type(x, ' derived_prop')
or type(x, ' trustRelationElm'))
^ not(exist(t1) (effectiveFrom(x,t1)))
-> effectiveFrom(x,-M))

Axiom DKP-3:

for-all(x,t2)
(((type(x, 'asserted_prop') or type(x, 'derived_prop')
or type(x, 'trustRelationElm'))
^ not(exist(t1) (effectiveTo(x,t2)))
-> effectiveTo(x,+M))

At a given time point, an asserted_prop is effective if the time point is within its effective period (time interval).

Axiom DKP-4:

for-all(t,x,t1,t2)
((type(x, 'asserted_prop')
^ effective_from(x,t1) ^ effective_to(x,t2) ^ t1 ≤ t ^ t ≤ t2)
-> effective_at(x,t))

At a given time point, a derived_prop is effective if: (1) its dependency KP_prop is effective at the time point; and (2) the time point is within the effective period of the derived_prop.

Axiom DKP-5:

for-all(t,x,y,t1,t2)
((type(x, 'derived_prop')
^ is_dependent_on(x,y) ^ effective_at(y,t)
^ effective_from(x,t1) ^ effective_to(x,t2) ^ t1 ≤ t ^ t ≤ t2)
-> effective_at(x,t))

At a given time point, an equivalent_prop / negative_prop is effective if its dependency node is effective at the time point.

Axiom DKP-6:

for-all (x, y, v, t)
((type(x, 'equivalent_prop') or type(x, 'negative_prop')))
^ is_dependent_on(x, y) ^ effective_at(y, t))

-> effective_at(x, t))

At a given time point, an and_prop is effective, if all its dependency KP_props are effective at the time point.

Axiom DKP-7:

for-all(x,t)
((type(x, "and_prop")
^ for-all (y)(is_dependent_on(x,y)
-> effective_at(y, t)))
->effective_at(x, t))

At a given time point, an or_prop is effective, if at least one of its dependency KP_props is effective at the time point.

Axiom DKP-8:

for-all(x,t)
((type(x, "or_prop")
^exist(y)(is_dependent_on(x,y) ^ effective_at(y, t)))
->effective_at(x, t))

Trusted at a time point

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

Axiom DKP-9:

for-all (a,x,fl,z,c,w,t)
((type(x, "KP-prop")
^ has_infoCreator(x, c) ^ in_fields(x, fl) ^ contained_in(z, fl)
^ trusted_in(a, c, w, t)
^ subfield_of(z, w)
^ has_sig_status(x, "Verified"))
->trusted(x, a, t)).

An information creator is trusted in a specific knowledge field at a given time point, if the time point is within the effective period of the trust relation element.

Axiom DKP-10:

for-all (a, c, f, t, t1, t2)
((trusted_in_during(a ,c, f, , t1, t2) ^ t1 ≤ t ^ t ≤ t2)
-> trusted_in(a,c,f,t))

The following axioms extend the axioms in static KP with the concept of "effective period".

Asserted Propositions

At a given time point, an asserted-prop has its truth value as assigned, if the asserted-prop is trusted by the provenance agent at the time point, and the proposition is effective at the time point.

Axiom DKP-11:

for-all (a,x,v,t)
((type(x, "asserted_prop")
^ trusted(x, a, t) ^ effective_at(x, t)
^ assigned_truth_value(x, v))
->trusted_truth_value(a, x, v, t)).

Equivalent Propositions

At a given time point, the trusted truth value of an equivalent-prop is the same as the trusted truth value of the proposition it depends on.

Axiom DKP-12:

for-all (a, x, y, v, t) ((type(x, "equivalent_prop")
^ trusted(x, a, t)
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, v, t))
->trusted_truth_value(a, x, v, t)).

Composite Propositions

At a given time point, the trusted truth value of a negative-prop is the negation of the trusted truth value of the KP-prop it is dependent on.

Axiom DKP-13

for-all (a, x, y, t)
((type(x, "negative_prop")
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, "True", t))
-> trusted_truth_value(a, x, "False", t)).

Axiom DKP-14:

for-all (a, x, y, t)
((type(x, "negative_prop")
^ is_dependent_on(x, y) ^ trusted_truth_value(a, y, "False", t))
-> trusted_truth_value(a, x, "True", t)).

Axiom DKP-15:

for-all (a, x, y, t)
((type(x, "negative_prop")
^ is_dependent_on(x, y)
^ trusted_truth_value(a, y, "Unknown", t))
->trusted_ truth_value(a, x, "Unknown", t)).

At a given time point, the trusted truth value of an and-prop is "True" if all its dependency KP-props are "True" at the time point; The trusted truth value of an and-prop is "False" if at least one of its dependency KP-props is "False"; and the trusted truth value of an and-prop is "Unknown" if at least one of its dependency KP-props is "Unknown" and none of them is "False".

Axiom DKP-16:

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

Axiom DKP-17:

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

Axiom DKP-18:

for-all(a, x, t)
((type(x, "and_prop")

^ *(exist(y) (is_dependent_on(x, y)*
^ trusted_truth_value(a, y, "Unknown")))
^ *(not ((exist y) (is_dependent_on(x, y)*
^ trusted_truth_value(a, y, "False", t)))))
->trusted_truth_value(a, x, "Unknown", t)).

At a given time point, the trusted truth value of an or-prop is "True" if at least one of its dependency KP-props is "True" at the time point; the trusted truth value of an or-prop is "False" if all its dependency KP-props are "False"; and the truth value of an or-prop is "Unknown" if at least one of its dependency KP-props is "Unknown" and none of them is "True".

Axiom DKP-19:
for-all(a, x, t)
((type(x, "or_prop")
^ (exist (y) (is_dependent_on(x, y)
^ trusted_truth_value(a, y, "True", t))))
->trusted_truth_value(a, x, "True", t)).

Axiom DKP-20:
for-all(a, x, t)
((type(x, "or_prop")
^ (for-all (y) (is_dependent_on(x, y)
^ trusted_truth_value(a, y, "False", t))))
->trusted_truth_value(a, x, "False", t)).

Axiom DKP-21:
for-all(a, x, t)
((type(x, "or_prop")
^ (not ((exist y) (is_dependent_on(x, y)
^ trusted_truth_value(a, y, "True", t))))
^ ((exist y) (is_dependent_on(x, y)
^ trusted_truth_value(a, y, "Unknown", t))))
->trusted_truth_value(a, x, "Unknown", t)).

Derived Propositions

At a given time point, the truth value of a derived proposition is "True" or "False" as assigned, if the derived_prop is effective, trusted, and its dependency KP-prop (premise) is "True".

Axiom DKP-22:
for-all (a, x, y, v, t)
((type(x, "derived_prop")
^ effective_at(x, t)
^ trusted(x, a, t) ^assigned_truth_value(x, v)
^ is_dependent_on(x, y)
^ trusted_truth_value(a, y, "True", t))
->trusted_truth_value(a, x, v, t)).

6. IMPELEMENTATION AND EXAMPLE

In order to use knowledge provenance to judge the validity of web information, three tasks need to be conducted: (1) for information creators to web documents with KP metadata. We define KP metadata using RDFS; (2) for a provenance requester (or information user) to define trust relations; (3) to develop online KP agent to conduct provenance reasoning on propositions contained in web documents by using KP axioms.

For the first step, we have implemented the Dynamic KP model with an experimental system, called RDFS-Prolog. The system reasons about RDFS data. In the system, all RDFS data are represented equivalently as triples in the form of rdf_triple(S, P, O) where S denotes "Subject", P denotes "Predicate", and O denotes "Object". The semantics of RDFS and axioms of KP are represented with Prolog rules. In this way, the system can infer the truth of any KP-prop.

The following example illustrates KP annotation in web documents and knowledge provenance reasoning.

Consider the example discussed in section 3 regarding an IT supply chain composed of a reseller (FS), a distributor (DT) and a manufacturer (HP). As shown in figure 1, the product management department of the distributor (DT) created a derived proposition, "We should order 8,000 desktop PCs configured with 3.06G Pentium 4 processor from HP" (effective from 2003-05-26 to 2003-05-31), which is dependent on two propositions: (1) an equivalent proposition created by the sales department of the distributor stating "There is an increasing market demand for desktops with 3.06G Pentium 4 processor", which is dependent on an asserted proposition with same proposition content created by a contracted reseller called FS; and (2) an assertion created by manufacturer HP that says that "10,000 desktop PCs configured with 3.06G Pentium 4 processor are available" (effective from 2003-05-26 to 2003-06-01).

KP Meta Annotation

The web document that contains the derived proposition and its dependency and-proposition created by the product management department can be embedded with KP metadata as follows. The annotation to other web documents is in a similar way. Rather than maintain provenance in separate "meta" documents, KP metadata is embedded directly in a web document containing the propositions, making it easier to read and maintain.

Document: http://www.pm.examp.com/doc4

```
<HTML xmlns="http://www.w3.org/1999/xhtml"
  dsig =  "http://www.w3.org/2000/09/xmldsig#"
  kp = "http://www.eil.utoronto.ca/kp#"
  xml:lang="en" lang="en">
<HEAD>
<kp:Document rdf:about="http://www.pm.examp.com/doc4#"/>
</HEAD>
<BODY>

<kp:derived_prop rdf:id="order_PCP4">
   <kp:proposition_content>
We should order 8,000 desktops configured with 3.06G Pentium
4 processor from HP
   </kp:proposition_content>
   <kp:assigned_truth_value>"True"</kp:assigned_truth_value>
   <kp:is_dependent_on>"#demand_supply_PCP4"
   </kp:is_dependent_on>
   <kp:infoSource>
      <kp:creator>"Product Management Department"
      </kp:creator>
   </kp:infosource>
   <kp:in_fields>"Supply"</kp:in_fields>
</kp:Derived_prop>

<kp:and_prop rdf:id="demand_supply_PCP4">
```

```
    <kp:is_dependent_on>
       "http://www.hp.examp.com/doc2#available_PCP4_HP"
    </kp:is_dependent_on>
    <kp:is_dependent_on>
       "http://www.hp.examp.com/doc3#demands_PCP4"
    </kp:is_dependent_on>
</kp:and_prop>

<Signature ID="ProdMgmt--order-PCP4">
   <SignedInfo>
      <CanonicalizationMethod
  Algorithm=
"http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <SignatureMethod
 Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
      <Reference URI="#order_PCP4">
          <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          <DigestValue>j6hm43k9j3u5903h4775si83...=
          </DigestValue>
      </Reference>
   </SignedInfo>
   <SignatureValue>M459ng9784t...</SignatureValue>
   <KeyInfo>
     <X509Data>
       <X509SubjectName>...</X509SubjectName>
       <X509Certificate>MIID5jCCA0+gA...lVN
       </X509Certificate>
     </X509Data>
   <KeyInfo>
 </Signature>
</BODY>
</HTML>
```

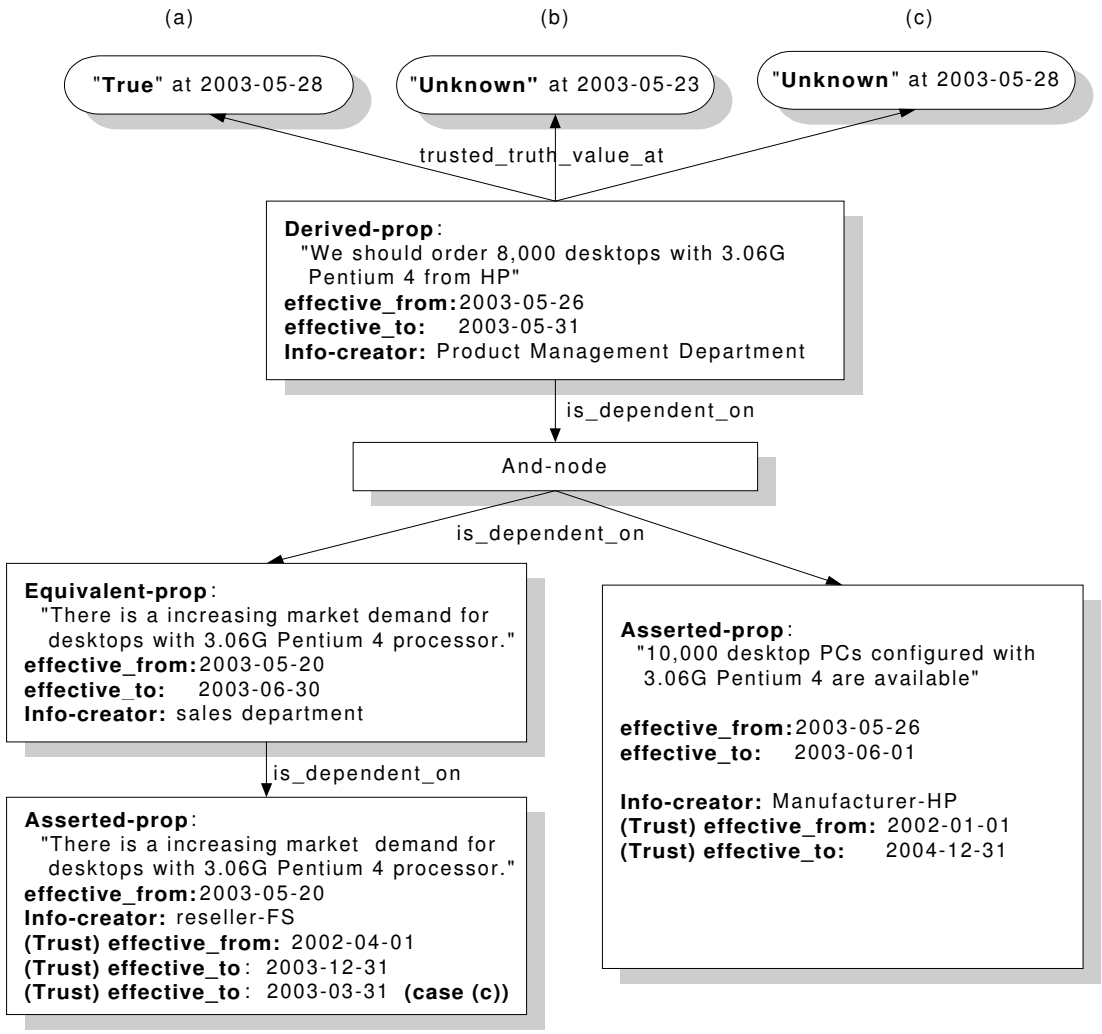

Figure 1. Application example of Knowledge Provenance

KP Reasoning

In the example, assume the trust relation effective periods (associated with contracts) are as follows: HP is trusted by DT from 2002-01-01 to 2004-12-31; FS is trusted by DT from 2002-04-01 to 2003-12-31; and in case (c) of figure 1, FS is trusted

from 2002-04-01 to 2003-03-31. In addition, assume the sales department and product management department are trusted within the distributor (DT) on topic {"Order", "Demands"} and {"Products", "Supply"} respectively.

As shown in figure 1, case (a) requests the trusted truth value of the derived proposition that "We should order 8,000 desktop PCs configured with 3.06G Pentium 4 processor from HP" at time point 2003-05-28. "True" is obtained by reasoning using KP axioms; case (b) requests the trusted truth value of the derived proposition at 2003-05-23. "Unknown" is obtained, for the reason that 2003-05-23 is not covered by [2003-05-26, 2003-05-31], the effective period of HP' s asserted proposition, which causes the asserted proposition and further the derived proposition are not effective; case (c) requests the trusted truth value of the derived proposition at 2003-05-28. "Unknown" is reached because 2003-05-28 is not covered by [2002-04-01,2003-03-31], the effective period of trust to reseller FS, which causes that FS cannot be trusted and further its assertion and all proposition dependent on it cannot be trusted also.

The major parts of the provenance reasoning for case (a) by using our RDFS-Prolog are listed as follows.

?- trusted_truth_value(' KP_agent' ,uri(doc4,' order_PCP4'),V, ' 20030528').

… (some intermediate outputs are omitted).

Axiom DKP-5 Output:
Derived-prop uri(doc4, order_PCP4) is effective at time point 20030528.
Because:
(1) the derivation is effective from 20030526 to 20030531;
(2) the dependency proposition uri(doc4, demand_supply_PCP4) is effective at the time point 20030528.
…

Axiom DKP-9 Output:
Proposition uri(doc4, order_PCP4) is trusted at time point: 20030528 by Agent: KP_agent.
Because:
(1) info-creator uri(doc4, Product Management Department) is trusted in the field of Products by the agent at the time point;
(2) digital signature is verified successfully.
…

Axiom DKP-16 Output:
Agent KP_agent trusts that and-prop uri(doc4, demand_supply_PCP4) has trusted truth value of True at time point: 20030528.
Because:
(1) uri(doc4, demand_supply_PCP4) is dependent on [uri(doc2, available_PCP4_HP), uri(doc3, demands_PCP4)]
(2) all of them have trusted truth value of True at the time point.

Axiom DKP-22 Output:
Agent KP_agent trusts that derived-prop uri(doc4, order_PCP4) has trusted truth value of uri(kp1, True) at time point: 20030528.
Because:
(1)uri(doc4, order_PCP4) is dependent on uri(doc4, demand_supply_PCP4)
(2)uri(doc4, demand_supply_PCP4) has trusted truth value of True.
V = uri(kp1, ' True')
Yes

7. Summary

Knowledge Provenance (KP) is proposed to address the problem of how to determine the validity and origin of web information. Four levels of KP comprising Static KP, Dynamic KP, Uncertain KP, and Judgement-based KP have been identified. This paper focuses on Dynamic Knowledge Provenance to address the problem of how to determine the validity of web information over time.

Dynamic Knowledge Provenance determines the validity of dependent information in a world where both the validity of the information it depends on and the trustworthiness of the information creators may change over time. We define an ontology, including terminology and semantics, and its implementation for the Semantic Web.

This research was supported, in part, by Bell University Laboratory.

8. REFERENCES

[1] Allen, J.F. and Ferguson, G., (1994), Actions and Events in Interval Temporal Logic, *J. Logic and Computation*, 4, 5, 1994.

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

[3] Bartel, M., J. Boyer, B. Fox, B. LaMacchia, E. Simon, (2002), XML-Signature Syntax and Processing, http://www.w3.org/TR/xmldsig-core/

[4] Berners-Lee, T., (2003), Semantic Web Status and Direction, ISWC2003 keynote. http://www.w3.org/2003/Talks/1023-iswc-tbl/

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

[6] FOAF, (2002), http://www.foaf-project.org/

[7] 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

[8] Gil, Y. and Ratnakar, V., (2002), "Trusting Information Sources One Citizen at a Time", *Proceedings of ISWC'02*.

[9] Golbeck, J., Hendler, J., and Parsia, B., (2002), Trust Networks on the Semantic Web, University of Mariland, College Park.

[10] 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.

[11] Huang, J., and Fox, M.S., (2003), "Uncertain Knowledge Provenance", *EIL Technical Report, University of Toronto*.

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

[13] Khare, R., and Rifkin, A., (1997), "Weaving and Web of Trust", *World Wide Web Journal,* Vol. 2, No. 3, pp. 77-112.

[14] Mayorkas, A. N., (2000), http://www.usdoj.gov/usao/cac/pr/pr2000/003.htm

[15] McGuinness, D.L., and Pinheiro da Silva, P., (2003), Infrastructure for Web Explanations, *ISWC'03*.

[16] Oliver, K., (1997), Evaluating the Quality of Internet Information. Virginia Tech.

[17] Pinto, J. and Reiter, R. (1993), Temporal Reasoning in Logic Programming: A Case for the Situation Calculus, *Proceedings of the Tenth International Conference on Logic Programming*. June 1993. pp 203-221.

[18] Richardson, M., Agrawal, R., and Domingos, P., (2003), Trust Management for the Semantic Web, *ISWC'03*, PP.351-368.

[19] 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

[20] Tan, K. (2002), Building Your Appropriate Certificate-based Trust Mechanism for Secure Communications, *White Paper,* March, 2002. http://www.rainbow.com/library/8/BuildingYourAppCert_based.pdf