Inference Web: Portable and Sharable Explanations for Question Answering

Deborah L. McGuinness and Paulo Pinheiro da Silva

Knowledge Systems Laboratory
Stanford University
Stanford, CA 94305
{dlm | pp}@ksl.stanford.edu

Abstract

The World Wide Web lacks support for explaining information provenance. When web applications return results, many users do not know what information sources were used, when they were updated, how reliable the source was, what information was looked up versus derived, and if something was derived, how it was derived. In this paper we introduce the Inference Web (IW) that addresses the problems of opaque query answers by providing sharable, combinable, and distributed explanations. The explanations include information concerning where answers came from and how they were deduced (or retrieved). The IW solution includes: an extensible registry containing details on information sources and reasoners, a portable proof specification, and an explanation browser.

Motivation

Inference Web (IW) aims to enable applications that can generate sharable and distributed explanations for any of their results. There are many reasons that users and agents need to understand the provenance of information that they get back from applications. The main motivating factors for us are interoperability, reuse, and trust. Interoperability is essential if agents are to collaborate. Trust and reuse of retrieval and deduction processes is facilitated when explanations are available. Ultimately, if users and/or agents are expected to trust information and actions of applications and if they are expected to use and reuse application results potentially in combination with other information or other application results, they may need to have access to many kinds of information such as source, recency, authoritativeness, method of reasoning, term meaning and interrelationships, etc.

In our work, we build on past experience designing explanation components for reasoning systems [McGuinness, 1996; McGuinness and Borgida, 1995; Borgida, et. al, 1999] and experience designing query components for frame-like systems [McGuinness, 1996; Borgida and McGuinness, 1996] to generate requirements. We also obtained input from contractors in three DARPA-sponsored programs concerning knowledge-based applications (the High Performance Knowledge Base program[1], Rapid Knowledge Formation Program[2], and the DARPA Agent Markup Language Program[3]). We also obtained requirements from literature on explanation for expert systems [Swartout, et. al., 1991] and usability of knowledge representation systems [McGuinness-Patel-Schneider, 1998 and 2003].

In this paper, we include a list of requirements gathered from past work and from surveying users. We present the IW architecture and provide a description of the major components including the portable proof specification, the registry (containing information about inference engines, proof methods, and ontologies), and the justification browser. We also provide some simple usage examples. We conclude with our contributions in the areas of application interoperability, reuse, and trust.

Requirements

If humans and agents need to make informed decisions about when and how to use answers from applications, there are many things to consider. Decisions will be based on the quality of the source information, the suitability and quality of the reasoning engine, and the context of the situation. First we will consider issues concerning the source information.

Even when applications such as search engines or database systems just look up asserted or "told" information, users (and agents) may need to understand where the source information came from at varying degrees of detail. Sometimes this information is called provenance and may be viewed as meta information about told information. Provenance information may include:

- Source name (e.g., CIA World Fact Book)
- Date and author(s) of last update
- Author(s) of original information

[1] http://reliant.teknowledge.com/HPKB/
[2] http://reliant.teknowledge.com/RKF/
[3] http://www.daml.org

- Authoritativeness of the source (is this knowledge store considered or certified as reliable by a third party?)
- Degree of belief
- Degree of completeness (Within a particular scope, is the source considered complete. For example, does this source have all of the employees of a particular organization up until a particular date? If so, not finding a particular employee would mean that they are not employed, counting employees would be an accurate response to number of employees, etc.)

The information above could be handled with meta information about content sources and about individual assertions. Additional types of information may be required if users need to understand the meaning of terms or implications of query answers. If applications make deductions or otherwise manipulate information, users may need to understand how deductions were made and what manipulations were done. Information concerning term meaning, derived or manipulated information may include:

- Term or phrase meaning (in natural language or a formal language)
- Term inter-relationships (ontological relations including subclass, superclass, part-of, etc.)
- The source of derived information (reasoner used, reasoner method, reasoner inference rule, etc.)
- Reasoner description (is the reasoner used known to be sound and complete?)
- Term uniqueness (is D.L. McGuinness the same individual as Deborah McGuinness?)
- Term coherence (is a particular definition incoherent?)
- Source consistency (is there support in a system for both A and ~A)
- Were assumptions used in a derivation? If so, have the assumptions changed?

Use Cases

Every combination of a query language with a query-answering environment is a potential new context for the Inference Web. We provide two scenarios as motivating use cases. Consider the situation where someone has analyzed a situation previously and wants to retrieve this analysis. In order to present the findings, the analyst may need to defend the conclusions by exposing the reasoning path used along with the source of the information. In order for the analyst to reuse the previous work, s/he will also need to decide if the source information used previously is still valid (and possibly if the reasoning path is still valid).

Another simple motivating example arises when a user asks for information from a web application and then needs to decide whether to act on the information. For example, a user might use a search engine interface or a more sophisticated query language such as DQL[1] for retrieving lookup information such as "red cashmere sweater" or "find a red cashmere sweater costing less than 200 dollars that is ready for immediate shipping". Moreover, the user might ask for an explanation along with answers since s/he may want information such as the data came from a reliable merchant and the data was updated in the last 24 hours.

In order for this scenario to be operationalized, we need to have the following:

- A way for applications (reasoners, retrieval engines, etc.) to dump justifications for their answers in a format that others can understand. To solve this problem we introduce a portable and sharable proof specification.
- A place for receiving, storing, manipulating, annotating, comparing, and returning meta information used to enrich proofs and proof fragments. In order to address this requirement, we introduce the Inference Web Registry for storing the meta information and the Inference Web Registrar web application for handling the Registry.
- A way to present justifications to the user. As one solution to this problem, we introduce a proof browser.

Inference Web

We begin with a short description of different categories of Inference Web users. These users along with the usage examples above motivate the three main components of Inference Web: portable proofs, registry, and proof browsers.

The prime users of inference web are:

- Application developers (authors of reasoners, search engines, database systems, etc.) who would like to defend why their answers to queries should be believed or who would like to state under what conditions their systems are best used.
- Authors of hybrid solutions programs interested in combining multiple answering systems and/or knowledge bases. These people need to understand how terms relate to each other and how answers were derived and might integrate. Examples of such people include ontology builders who are merging ontologies or extending ontologies, crawler or wrapper authors, people combining databases or knowledge based systems, etc.
- Human or agents needing to decide if they can trust either retrieved information or inference processes used to retrieve information. This user may view partial or complete justifications for answers.

Portable Proofs: Systems that may be asked to return a justification for an answer along with an answer need to expose provenance information along with their deductive process including possibly meta information about the

[1] http://www.daml.org/2002/08/dql/ .

system itself. We provide a specification in written in the web markup language DAML+OIL [Dean, et. al, 2002].

Our portable proof specification includes the four major components of IW proof trees: inference rules, inference steps, well formed formulae (WFFs), and ontologies. Inference rules (such as modus ponens) can be used to deduce a consequent (a well formed formula) from any number of antecedents (also well formed formulae). An inference step is a single application of an inference rule. The inference step will be associated with the consequent WFF and it will contain pointers to the antecedent WFFs, the inference rule used, and any variable bindings used in the inference rule application. The antecedent WFFs may come from other inference steps, existing ontologies, extraction from documents, or they may be assumptions.

A proof can then be defined as a tree of inference steps explaining the process of deducing the consequent WFF. In Inference Web, proofs are *trees of proof fragments* rather than single monolithic proofs. IW proofs are proof fragments since the last inference step used to derive a WFF can be presented alone with links to its antecedents and variable bindings. Asking how each antecedent WFF was derived generates follow-up questions. These individual proof fragments may be composed together to generate a complete proof, i.e., a set of inference steps that have no antecedents that are derived rather than asserted information.

A WFF may be the consequent of any number of inference steps. This can be used to support multiple justifications for any particular WFF. WFFs may not be the consequent of any inference step if they are assumptions or merely asserted information in an ontology that the user is referencing. The specification of IW concepts is available at http://www.ksl.stanford.edu/software/IW/spec.

Registry: The IW registry is currently a centralized repository of information used to enrich explanations with details about *authoritative sources*, *ontologies*, *inference engines,* and *inference rules*. Our registry includes template information about each of the classes in the registry. For example, inference engines may have the following properties associated with them: name, URL, author(s), date, version number, organization, etc. The current demonstration registry is available at: http://belo.stanford.edu:8080/iwregistry/BrowseRegistry.jsp.

Information in the registry contains the information linked to in the proofs. Every inference step should have a link to at least one inference engine that was responsible for instantiating the inference step itself, as can be observed in Figure 1.

The description of inference rules is one of the most important features of the Registry. Registered rules can be atomic or can be derived from other registered rules. Thus, reasoner-specific rules can be explained in the Registry before the reasoner is actually used to generate IW proofs. Inference web thus provides a way to use one reasoner to explain another reasoner's inference rules. This may be useful for explaining heavily optimized inference engines. Inference web's registry, when fully populated, will contain inference rule sets for many common reasoning systems. Users may view inference rule sets to help them decide whether to use a particular inference engine.

```
<?xml version='1.0'?> <rdf:RDF (...)>
<iw:WFF>
    <iw:WFFContent> (a WFF is stored as a predicate logic
                     conjunctive normal-form sentence)
      <daml:List
rdf:about='IW/spec/fopl.daml#Clause'>
        <daml:first>
          <fopl:Negated-Predicate-Of-Terms
            fopl:SymbolName='holds'>
          <fopl:hasArgumentList
rdf:parseType='daml:collection'>
            <iw:Constant>
<fopl:SymbolName>type</fopl:SymbolName>
</iw:Constant>
            <fopl:Variable fopl:SymbolName='?inst'/>
           (...)
      </daml:List>
    </iw:WFFContent>
    <iw:isConsequentOf
rdf:parseType='daml:collection'>
     (a WFF can be associated to a set of Inference steps)
     <iw:InferenceStep>
        <iw:hasInferenceRule
                rdf:parseType='daml:collection'>
          <iw:InferenceRule
                rdf:about='../registry/IR/GMP.daml'/>
        </iw:hasInferenceRule>
        <iw:hasInferenceEngine
                rdf:parseType='daml:collection'>
           <iw:InferenceEngine
                rdf:about='../registry/IE/JTP.daml'/>
         </iw:hasInferenceEngine>
           (...)
        <iw:has Antecedent
                rdf:parseType='daml:collection'>
        (inference step antecedents are IW files with
         their own URIs)
          <iw:WFF rdf:about='../sample/IW3.daml'/>
          <iw:WFF rdf:about='../sample/IW4.daml'/>
        </iw:hasAntecedent>
        <iw:hasVariableMapping
rdf:type='http://www.daml.org/2001/03/daml+oil#List'/
>
           (...)
      </iw:InferenceStep>
    </iw:isConsequentOf>
  </iw:WFF>
</rdf:RDF>
```

Figure 1 An Inference Web proof.

Browser: We provide an Inference Web browser that displays proof fragments to end users in a number of formats. Initially, we include a limited English form, KIF[1], and conjunctive normal form. We also expect that some applications may implement their own displays using our API.

The browser implements a lens metaphor responsible for rendering a fixed number of levels of inference steps depending on the lens magnitude. The default lens number is one, thus the browser displays the inference step used to derive it including its antecedents.

[1] http://logic.stanford.edu/kif/kif.html .

We believe that one of the keys to presentation of justifications is breaking proofs into separable pieces. Since we present fragments, automatic follow-up question support is a critical function of the IW browser. Every element in the viewing lens can trigger a browser action. The selection of an antecedent that is derived re-focuses the lens on an antecedent's inference step. For other lens elements, associated actions present Registry meta-information in the Trust Disclosure Panel. The selection of the consequent presents details about the inference engine used to derive the actual theorem. The selection of an inference rule presents a description of the rule. The selection of an axiom presents details about ontologies where the axiom is defined. An example of this process can be seen from the Inference Web web pages at: http://www.ksl.stanford.edu/software/iw/Ex1/.

Contributions and Future Work

Inference Web provides the following contributions:

- An architecture supporting interoperability between agents using portable proofs. Portable proofs are specified in the emerging web standard DAML+OIL so as to leverage XML-, RDF-, and DAML-based information services. Proof fragments as well as entire proofs may be interchanged.
- Lightweight proof browsing using the lens-based IW proof browser supporting either pruned justifications or guided viewing of a complete reasoning path.
- Support for alternative justifications using multiple inference steps. This can allow derivations to be performed by performance reasoners with explanations being generated by alternative reasoners more aimed at human consumption.
- Registry of inference engines, ontologies, and sources.

We are currently extending the Stanford's JTP[1] theorem prover to produce portable proofs and simultaneously populating the IW registry with JTP information. Future work includes expanding to include other inference engines. We also intend to provide specialized support for why-not questions expanding upon [Chalupsky-Russ,2002] and [McGuinness,1996]. We are also looking at additional support for proof browsing and pruning.

Conclusion

Inference web enables applications that can generate portable and sharable explanations of their conclusions. We described the three major components of IW – the portable proof specification based on the emerging web language-DAML, the registry of inference engine, inference rule, and ontology information, and the IW proof browser. We have implemented the IW approach for one inference engine(JTP) and we encourage additional users.

[1] http://www.ksl.stanford.edu/software/jtp/ .

Acknowledgements: We have received valuable input from many colleagues including in particular R. Fikes, G. Frank, J. Jenkins, and previously Y. Li from Stanford. Additionally we obtained valuable comments on our specification from many including H. Chalupsky, P. Clark, K. Forbus, and K. Murray.

References

Borgida, A., E. Franconi, I. Horrocks, D. McGuinness, and P. Patel-Schneider. ``Explaining ALC subsumption" Proceedings of the International Workshop on Description Logics(DL-99), Linköping, Sweden, July 1999, pp 33-36.

Chalupsky, H. and T. Russ. WhyNot: Debugging Failed Queries in Large Knowledge Bases. In *Proceedings of the Fourteenth Innovative Applications of Artificial Intelligence Conference (IAAI-02)*, pages 870-877.

Dean, M., D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider, and L. Andrea Stein. OWL Web Ontology Language 1.0 Reference. World Wide Web Consortium (W3C) Working Draft 29 July 2002. Latest version is available at http://www.w3.org/TR/owl-ref/ .

Richard Fikes, Pat Hayes, Ian Horrocks, editors. *DAML Query Language (DQL) Abstract Specification.* http://www.daml.org/2002/08/dql/ .

McGuinness, D. 1996. *Explaining Reasoning in Description Logics.* Ph.D. Thesis, Rutgers University, Technical Report LSCR-TR-277.

McGuinness, D. and Borgida, A. *Explaining Subsumption in Description Logics.* In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada.

McGuinness, D. and P. Patel-Schneider. ``From Description Logic Provers to Knowledge Representation Systems". Franz Baader, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, 2003.

McGuinness, D. and P. Patel-Schneider. ``Usability Issues in Knowledge Representation Systems". In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, July, 1998. Updated version of ``Usability Issues in Description Logic Systems" published in *Proceedings of International Workshop on Description Logics*, Gif sur Yvette, (Paris), France, September, 1997.

Smith, M., D. McGuinness, R. Volz and C. Welty. Web Ontology Language (OWL) Guide Version 1.0. World Wide Web Consortium (W3C) Working Draft. Latest version is available at http://www.w3.org/TR/owl-guide/ .

Swartout, W., C. Paris, and J. Moore. **"**Explanations in Knowledge Systems: Design for Explainable Expert Systems". In IEEE Intelligent Systems, June 1991. http://www.computer.org/intelligent/ex1991/x3058abs.htm**.**