Explanations in Dialogue Systems through Uncertain RDF Knowledge Bases

Daniel Sonntag and Martin Theobald

German Research Center for AI (DFKI) and Max Planck Institute for Informatics (MPII), Stuhlsatzenhausweg, 66123 Saarbruecken, Germany

Abstract. We implemented a generic dialogue shell that can be configured for and applied to domain-specific dialogue applications. The dialogue system works robustly for a new domain when the application backend can automatically infer previously unknown knowledge (facts) and provide explanations for the inference steps involved. For this purpose, we employ URDF, a query engine for uncertain and potentially inconsistent RDF knowledge bases. URDF supports rule-based, first-order predicate logic as used in OWL-Lite and OWL-DL, with simple and effective top-down reasoning capabilities. This mechanism also generates explanation graphs. These graphs can then be displayed in the GUI of the dialogue shell and help the user understand the underlying reasoning processes. We believe that proper explanations are a main factor for increasing the level of user trust in end-to-end human-computer interaction systems.

1 Introduction

Multimodal interaction with large and dynamic data repositiories is an important topic for the next generation of human-computer interaction systems. Over the last several years, we have focused on the idea that Semantic Web [Fensel et al., 2003] data structures provide new opportunities for *semantically-enabled user interfaces*. The explicit representation of the *meaning* of data allows us to (1) transcend traditional keyboard and mouse interaction metaphors, and (2) provide representation structures for more complex, collaborative interaction scenarios with more complex result presentations. Over the last years, we have adhered strictly to the developed rule “No presentation without representation,” in order to log the state of the dialogue system, the displayed elements, and the user queries explicitly.

On the database side, a new area of information retrieval (IR) has begun with the advent of structured databases in Semantic Web RDF structures and respective query languages. The dominant query language for these RDF[1] repositories is the W3C recommendation SPARQL[2]. For the next generation of human-computer interaction systems, explanation-based inference during data retrieval and uncertain knowledge plays a major role. In order to implement these properties, we use an efficient reasoning framework for graph-based, non-schematic RDF knowledge bases and SPARQL-like queries, Uncertain RDF (URDF). URDF augments first-order reasoning by combining

[1] See http://www.w3.org/TR/rdf-primer/ and http://www.w3.org/TR/rdf-schema/.
[2] http://www.w3.org/TR/rdf-sparql-query

soft rules, with Datalog-style recursive implications, and hard rules, in the shape of mutually exclusive sets of facts. It incorporates the common possible world semantics with independent base facts as it is prevalent in current probabilistic database approaches. And it supports semantically more expressive, probabilistic first-order representations, like, for example, Markov Logic Networks.

In this paper, we discuss a prototype system which provides a dialogue-based interaction with such a probabilistic advanced database while following the URDF model. URDF allow us to infer and present uncertain knowledge from Semantic Web databases in the multimodal dialogue context. More precisely, we gain graph-based knowledge for dialogue-based explanations with confidences for inferred knowledge. While incorporating the URDF functionality into a dialogue system, we provide a first prototype implementation of an explanation-aware multimodal dialogue system (as implemented in the Comet music retrieval system [Sonntag et al., 2009] and in the dialogue system for the medical domain [Sonntag and Möller, 2009], but without URDF and explanation knowledge). Such dialogue systems can answer complex questions and provide additional multimedia material such as graphs or videos.

Generally speaking, dialogue-based question answering (QA) allows a user to pose questions in natural speech, followed by answers presented in a concise textual form with multimedia material [Sonntag et al., 2007]. For example, "Who is the German Chancellor?" The short answer is "Angela Merkel" accompanied by a picture. The user, however, should not only be presented the factual answers to such questions, but also some explanations about the actual QA process. As [Glass et al., 2008] show, proper explanations are one main factor that influences the level of user trust in complex (adaptive) artificial intelligence systems. Deriving and using explanations in dialogue-based QA is a unique opportunity for enhancing trust especially in uncertain, inferred answers in human-computer interaction systems.

The paper is structured as follows: Section 2 discusses related work and Section 3 explains the dialogue system framework. The URDF framework is presented in Section 4, followed by an example dialogue (Section 5) and our conclusions (Section 6).

2 Related Work

Prominent examples of dialogue platforms include OOA [Martin et al., 1999], TRIPS [Allen et al., 2000], and Galaxy Communicator [Seneff et al., 1999]; these infrastructures mainly address the interconnection of heterogeneous software components. A comprehensive overview of ontology-based dialogue processing and the systematic realisation of these properties can be found in [Sonntag, 2010], pp.71-131. Many systems are available that translate natural language input into structured ontological representations (e.g., AquaLog [Lopez et al., 2005]), port the language to specific domains, e.g., ORAKEL [Cimiano et al., 2007], or use reformulated semantic structures NLION [Ramachandran and Krishnamurthi, 2009]. AquaLog, for example, presents a solution for a rapid customisation of the system for a particular ontology; with ORAKEL a system engineer can adapt the natural language understanding (NLU) component in several cycles thereby customising the interface to a certain knowledge domain; and NLION uses shallow natural language processing techniques (i.e., spell checking, stemming, and compound detection) to realise a single semantic concept or an ontology property. All of them support the translation to SPARQL queries in principal. However, all of them deal with written keywords or simple semantic relations, e.g., X *isDefinedAs* Y. They do not focus on the much more complex explanation-based answering process while using a dialogue system.[3]

As introduced, the dominant query language for RDF repositories is the W3C recommendation SPARQL. Similar RDF-based query languages are worth mentioning, such as RDQL[4] or SERQL[5]. These languages are based on the notion of RDF triple patterns, which can be connected via several query operators such as "union" or "filter". In previous implementations of dialogue system backends without URDF, we used SPARQL queries because they are the de facto standard. We also used the resources in the Linked Data framework (see [Bizer, 2009]). The RDF triple structure itself, which is used in Linked Data, represents enough structure to be called a database index, which maps a wildcard triple pattern onto the matching concrete data triples. Although the Linked Data sources are updated frequently, they can be considered rather static, i.e., OWL-style reasoning about these sources is normally not provided. Moreover, the SPARQL 2 specification (which most of these endpoints implement) provides support for operators such as "group by" or aggregate functions (e.g., COUNT, MIN, MAX, SUM). In the context of unstructured natural language input, SPARQL also provides convenient operator extension, i.e., the "filter" operator, to specify free test searches and even regular expressions based on operations for regular expressions. Examples of how these operators can be used in the context of integrating Linked Data for semantic dialogue and backend access can be found in [Sonntag and Kiesel, 2010].

3 Dialogue System Framework

In earlier projects [Wahlster, 2003,Reithinger et al., 2005] we integrated different subcomponents to multimodal interaction systems. In the context of the new THESEUS programme[6], we then implemented a situation-aware dialogue shell for semantic access to image media, their annotations, and additional textual material. We use a distributed, ontology-based, dialogue system architecture, where every major component for speech understanding, dialogue management, or speech synthesis can be run on a different host, increasing the scalability of the overall system.[7] A shared representation and a common knowledge base ease the dataflow within the system and avoid costly and error-prone transformation processes (c.f. "No presentation without representation"). More precisely, an ontology-based representation of a user query can be used to create a query that can be posed to a (U)RDF endpoint.

[3] In addition, these systems directly transfer the input to the desired SPARQL queries without dealing with the complex influences of message passing in dialogue frameworks or input fusion.

[4] http://www.w3.org/Submission/RDQL/

[5] http://www.openrdf.org/doc/sesame/users/ch06.html

[6] http://www.theseus-programm.de

[7] The dialogue system architecture is based on a generic framework for implementing multimodal dialogue systems (ODP platform, available at http://www.semvox.de/).

The dialogue system acts as the middleware between the clients and the backend services (i.e., the RDF repositories with an online API) that hide complexity from the user by presenting aggregated ontological data. Figure 1 shows the dialogue system architecture. The client provides means to connect to the dialogue system via the *event bus*, to notify it of occurred events, to record and play back audio streams, and to render the received display data obtained from the *dialogue system/dialogue manager*. The generated (U)RDF queries are then processed by the backend system (Remote RDF Repository) in order to retrieve the requested entities.

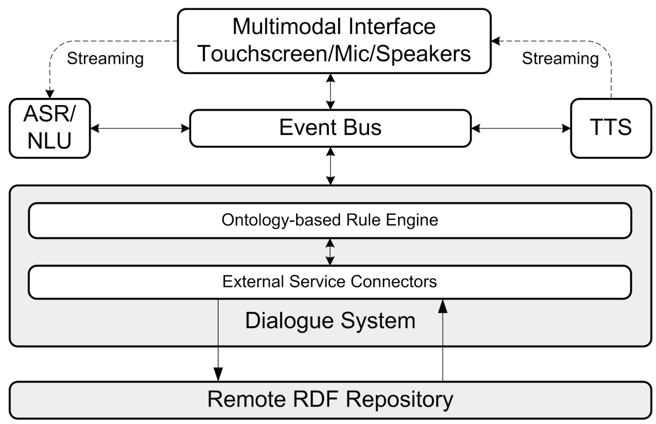

Fig. 1: Dialogue System Architecture

A central building block for component development and an integral part of the dialogue middleware is the included application programming interface for the efficient representation of ontology-based data using extended Typed Feature Structures (eTFS). As described in [Schehl et al., 2008], the eTFS API is tightly integrated into a production rule system which enables a declarative specification of the processing logic in terms of production rules. While processing the user input in the dialogue system, the output of the fusion step[8] is transferred to the backend system. Figure 2 provides a high-level view and rough sketch of the basic processing chain within the typical QA process. In this work, the backend access has been extended by addressing URDF. We use an Apache Tomcat server[9] for this purpose. The presentation has to be adapted to the result of the URDF process, i.e., graph-based explanations.

[8] A modality fusion component keeps track of the ongoing discourse, completes different types of anaphora, and merges input from different modalities. We use a production rule system, FADE, which is part of the ODP distribution.

[9] http://tomcat.apache.org/

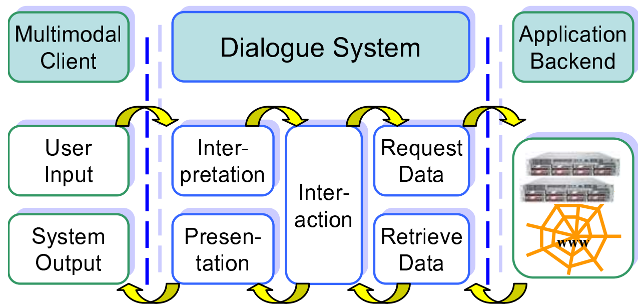

Fig. 2: Basic building blocks and core workflow of multimodal dialogue processing

4 URDF Framework

The URDF project, which is currently under development at the Max Planck Institute for Informatics, aims to enhance SPARQL-style query processing over RDF knowledge bases with simple and effective, top-down reasoning capabilities. Specifically, URDF supports rule-based, first-order reasoning concepts known from OWL-Lite and OWL-DL, thus capturing a decidable subset of first-order predicate logic. Moreover, since information extraction on the Web is often an iterative and inherently noisy process, URDF explicitly targets the resolution of *inconsistencies* between the underlying RDF facts and the inference rules. URDF also augments first-order reasoning by combining *soft rules*, which may be violated by some instances (facts) in the knowledge base, and *hard rules*, which may not be violated by any instance and can therefore be employed as hard consistency constrains, e.g., for capturing functional properties in OWL-Lite. Key to our approach for reasoning over uncertain data and resolving inconsistencies directly at query time is a novel and efficient approximation algorithm for a generalised version of the Weighted MAX-SAT problem, which allows URDF to dynamically cope with noisy data and/or evolving knowledge bases and changing domain constraints (rules). A further key feature of URDF is the ability to capture the resolution steps employed to infer answers by the reasoner in the form of an acyclic derivation graph over grounded rules (aka data "provenance" [Buneman and Tan, 2007] or "lineage" [Benjelloun et al., 2008]). This directed acyclic graph (DAG, cf. answer graphs) structure connects both base and derived facts via the rules that were used for grounding, which can be used to *explain* the answers given to a query and represent this explanation in graphical form to the user.

4.1 Representation Model and Expressiveness

URDF considers a *knowledge base* $\mathcal{KB} = \{\mathcal{F}, \mathcal{C}, \mathcal{S}\}$ as a triple consisting of RDF base facts $\mathcal{F}$, soft clauses $\mathcal{C}$, and hard (i.e., strict) rules $\mathcal{S}$. An *RDF graph* is a directed, labeled multi-graph, in which nodes are entities (such as individuals and literals), and labeled edges represent relationships between the entities. For example, an RDF graph can have an edge between the entity *Ullman* and the entity *Stanford*. This edge would be labeled with the relation name *worksAt*. More formally, an RDF graph over a finite set of relations Rel and a finite set of entities $Ent \supseteq Rel$ is a set of triplets (or facts) $\mathcal{F} \subset (Rel \times Ent \times Ent)$. RDF allows two entities to be connected by multiple relations (e.g., two people can be colleagues and friends at the same time). Thus, facts express binary relationships between entities. For readability, we will write a fact *(x,r,y)* in common prefix notation as *r(x,y)*.

As opposed to OWL, RDF cannot directly express relationships between facts (i.e., "facts over facts"). Relationships with higher arity can however be represented by introducing an *event entity*, i.e., a new entity that stands in binary relationships with all arguments of the n-ary fact. Alternatively, n-ary relationships can be represented using *reification* [Suchanek et al., 2008]. In RDF graphs, there is a distinction between individual entities (such as *Albert Einstein*) and class entities (such as the class *physicist*). Individuals are linked by the *type* relationship to their class. For example, *Albert Einstein* is linked to the class *physicist* by a statement *(AlbertEinstein, type, physicist)*. The classes themselves form a hierarchy. More general classes (such as *scientist*) include more specific classes (such as *physicist*). This hierarchy is expressed in RDF by edges with the *subclassOf* relationship: *(artist, subclassOf, singer)*.

Soft Rules We consider first-order logic rules over RDF facts. A *grounded soft rule* over a set $\mathcal{F}$ of RDF facts is a set $C \subseteq \mathcal{F}$ of facts, where each atomic fact $f \in C$ is marked as either positive or negative and thus becomes a *literal*. For example, a grounded rule could be:

$$\{\neg \textit{worksAt(Ullman, Stanford)},\ \textit{livesIn(Ullman, Stanford)}\}_{[0.4]}$$

Each soft rule is assigned a non-negative, real-valued weight. A higher weight indicates that matching the rules is of higher importance than matching a rule with a lower weight. To simplify talking about grounded rules of the same shape, we introduce *non-grounded rules*. A non-grounded rule C' is a grounded rule C over a set of facts in $\mathcal{F}$, where one or more entities are replaced by variables. A non-grounded rule C' over $\mathcal{F}$ implicitly stands for all grounded rules C that can be obtained by substituting the variables in C' by entities. Thus, the following rule subsumes the aforementioned grounded rule:

$$\{\neg \textit{worksAt(Ullman, x)},\ \textit{livesIn(Ullman, x)}\}_{[0.4]}$$

When grounded, the weight of the ungrounded rule is propagated to all its groundings. We use non-grounded rules solely to increase readability. We allow only *Horn rules*, i.e., rules where at most one literal is positive. Horn rules with exactly one positive literal can equivalently be rewritten as implications, in which all literals are positive. When written as implication, the *body* of a rule is a conjunction and the *head* consists of a single literal. In a first-order representation, only simple literals with no nested predicates or function calls are allowed in the rules. We can, however, extend the expressiveness of our reasoner (and yet remain in first-order) by allowing also rules with simple *arithmetic predicates*, which are "closed" within the rule, i.e., they can be evaluated (and thus be grounded) on-the-fly from the given variable bindings when the rule is processed. A grounded soft rule corresponds to a disjunction of literals, a so-called *clause*.

Hard Rules Hard rules are a distinct set of rules which define *mutually exclusive* sets of facts. Similarly to soft rules, hard rules can be expressed both in grounded and non-grounded form. A *grounded hard rule* is a set of facts $S \subseteq \mathcal{F}$ (also called a *competitor set*) that enforces the following constraint: a possible world $p : \mathcal{F} \to \{true, false\}$ can assign $true$ to at most one fact $f \in S$. For example, the following hard rule

{ *bornIn(Angela_Merkel, Hamburg)* ,
bornIn(Angela_Merkel, München) ,
bornIn(Angela_Merkel, Stuttgart) } ■

specifies that *Angela Merkel* could be born in at most one out of the above cities. Similarly to soft rules, we introduce *non-grounded hard rules*, where constants may be replaced by variables. For example, *bornIn(Angela_Merkel,x)* may be used to mark all the possible birth places of *Angela Merkel* in the knowledge base as mutually exclusive. Hard rules may not be violated and thus have no weights assigned (hence they are marked by a ■). For expressing these mutual-exclusion constraints, the hard rules encode special Horn clauses with only negatived literals. Equivalently, they can be rewritten as a number of conjunctions over binary Horn clauses with pair-wisely negated literals.

4.2 Reasoning Framework

The URDF reasoning framework combines classic first-order reasoning with a generalised Weighted MAX-SAT solver over both soft and hard rules. Query processing with URDF consists of two phases: 1) lookups of basic query patterns against the knowledge base, which involves both direct lookups of base facts in the knowledge base, but also recursively grounding rules and inferring new facts; and 2) resolving potential inconsistencies by a second reasoning step in the form of a Weighted MAX-SAT solver, which yields the final truth assignments to candidate answers obtained from the previous grounding step. That is, given a query in the form of a set of non-grounded atoms, we aim to find an assignment of truth values to the grounded query atoms (and all other grounded facts that are relevant for answering the query), such that the sum of the weights over the satisfied soft rules is maximised, without violating any of the hard constraints.

SLD Resolution and Dependency Graph Construction In the absence of any rules, URDF conforms to a standard (conjunctive) SPARQL engine, with the returned facts consisting only of grounded query atoms over base facts $\mathcal{F}$. URDF, however, allows for the formulation of recursive rules (i.e., with the same predicate occurring in the head as well as in the body of a rule), as well as mutually recursive sets of rules (i.e., with one rule producing grounded facts as input to another rule). Rather than reasoning about all facts in the knowledge base (which would be infeasible at query time), URDF investigates efficient top-down resolution algorithms for grounding rules against the knowledge base. Instead, we compute the so-called *dependency graph*, which consists only of facts in the knowledge base which are relevant for answering the query (including lineage pointers to grounded rules for the derived facts). Dependency graph construction is performed via SLD resolution [Apt and van Emden, 1982], which is similar to the resolution strategy used in Prolog and Datalog. Furthermore, SLD resolution over soft rules is also extended by a separate grounding phase for the hard rules (see [Theobald et al., 2010] for algorithmic details).

Resolving Inconsistencies After dependency graph construction, URDF constructs a propositional Boolean formula in conjunctive normal form (CNF) from all the lineage pointers to the grounded soft and hard rules, as well as the base facts in $\mathcal{F}$ used for grounding the rules. Since all rules are readily available in Horn clause form, the CNF can efficiently be constructed as a conjunction of all rules which are embedded in the dependency graph after the grounding phase. URDF employs an efficient approximation algorithm for a variant of the well-known Weighted MAX-SAT problem, which is specifically tailored to our setting, i.e., by considering a generalisation that is able to capture the presence of both soft rules (which may be violated in the MAX-SAT solution) and hard rules (which may not be violated by the solution). The MAX-SAT solver finally assigns *true* to only a subset of answer facts which are free of inconsistencies.

5 Example Dialogue

The following dialogue illustrates a user's practical interest in using a dialogue interface on top of a semantic URDF search engine for answering natural language questions. The dialogue concentrates around the questions about the URDF contents, i.e., factoid questions about celebrities, and the multimodal presentation of answer content. We use a big touchscreen installation for the presentation of the speech-based user requests (similar to our installation in [Sonntag et al., 2009]). On the touchscreen (cf. Figure 3), we display so-called semantic interface elements (SIEs). A SIE is a window on the GUI which displays aggregated multimedia results. For example, the Video SIE (Figure 3, left) displays videos from a YouTube API[10].

1 **U:** "Where is Angela Merkel born?"
2 **S:** Shows corresponding result in a SIE.
3 **U:** "What do Angela Merkel and Al Gore have in common?"
4 **S:** Shows corresponding relation graph.
5 **U:** "Where does he live?"
6 **S:** Shows corresponding relation graph.
Synthesises a summary of the graph's interpretation.

In the example dialogue, turn (1) results in a structured result display (Figure 4, "Hamburg" is also synthesised) according to factoid QA paradigm. The answer can be looked up directly in the URDF database (only hard rules apply in case of inconsistent knowledge). Turn (3) results in the display of a relation graph (similar to the graph in Figure 3, right). The last user turn (5) "Where does he live?" is of particular interest

[10] http://code.google.com/apis/youtube/overview.html

Fig. 3: Touchscreen surface with several semantic interface elements (SIEs)

from both a linguistic and database standpoint. We interpret the utterance as a deictic one, where the determination of the (celebrity) referent is dependent on the context in which it is said. Here, of course, the context is the referent "Al Gore", stored in the discourse context. In addition, the result is a complex explanation graph (Figure 5) derived from soft rules.

In the explanation (lineage) of the answer *bornIn(Al_Gore, Washington_D.C.)*, we can see that this fact could be derived from two different soft rules:

$$C_1 : \ \mathit{livesIn(a,b)} \leftarrow \mathit{marriedTo(a,c)} \wedge \mathit{livesIn(c,b)}$$

$$C_2 : \ \mathit{livesIn(a,b)} \leftarrow \mathit{marriedTo(a,c)} \wedge \mathit{bornIn(a,b)} \wedge \mathit{bornIn(c,b)}$$

The two derivations of the fact *bornIn(Al_Gore, Washington_D.C.)* are therefore connected by an OR-node in the graph which denotes a disjunction between the two subgraphs (while literals in the body of a rule would be considered conjunctive). The first derivation (in grey) denotes that Al Gore likely lives in Washington, D.C., because he is married to Tipper Gore, and rule C_1 expresses that married couples likely live at the same place. The place where Tipper Gore lives, on the other hand, also is not directly known in the knowledge base but is derived from similar inference steps and further groundings steps of different rules deployed in URDF. Overall, this subgraph reaches a recursion level of depth 4 for the inference about where Al Gore might actually live. The second derivation (in red) however shows that there is also a much shorter way of deriving the place where Al Gore lives, namely via rule C_2 which expresses that if two people are married and both were born in the same place (or area), then the former person also likely lives in the same place (or area). In other words, people might have strong ties to their birth place, which clearly is a form of "soft" inference. In this latter case, all the grounded facts that imply *bornIn(Al_Gore, Washington_D.C.)* can directly be grounded against base facts in the knowledge base in just a single inference step.

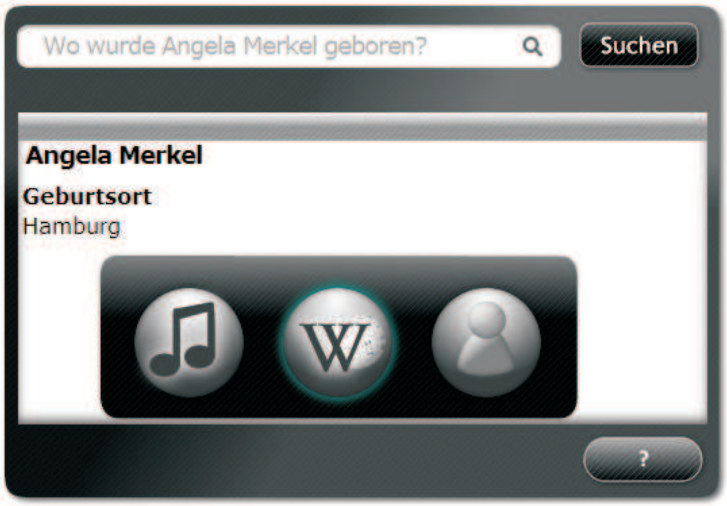

Fig. 4: Factoid answer SIE for the question “Where is Angela Merkel born?”

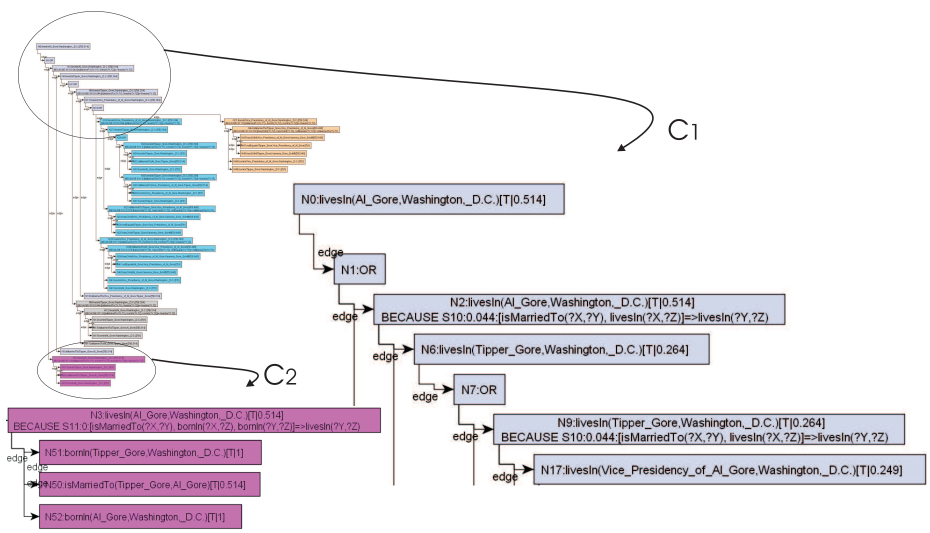

Fig. 5: Complex explanation graph for the question “Where does he (Al Gore) live?”

6 Conclusion

We have discussed explanations in dialogue systems through Uncertain RDF knowledge bases and presented URDF, a query engine for uncertain and potentially inconsistent RDF knowledge bases. This new backend can be integrated into a speech-based dialogue system to answer questions about a specific domain. Whereas factoid questions can be answered by state-of-the-art backend repositories via SPARQL queries, URDF provides the unique opportunity to also reason about uncertain knowledge and provide explanation graphs in the context of multimodal QA.

In our multimodal application scenario, a more complex and tighter integration of the provided result graphs has to be investigated. By extending the functional dialogue shell modules for a more complex dialogue behaviour on the result structures, we should be able to not only display the result graphs, but to paraphrase the result contents in natural language form as well. For example, the result graph could be presented in conjunction with the speech synthesis "I think he lives in Washington, D.C., because his wife, Tipper Gore, also lives there." This would, however, heavily exceed our current natural language generation capabilities, but pave the way toward speech-based explanations, a main factor for increasing the level of user trust in end-to-end human-computer interaction systems. Automatically inferred knowledge by URDF provides a new data stream and explanations for future, artificial intelligence based interaction systems.

Acknowledgments

This research has been supported in part by the THESEUS programme funded by the German Federal Ministry of Economics and Technology (01MQ07016). Thanks go out to Robert Nesselrath, Yajing Zang, Matthieu Deru, Simon Bergweiler, Gerhard Sonnenberg, Norbert Reithinger, Gerd Herzog, Alassane Ndiaye, Tilman Becker, Norbert Pfleger, Alexander Pfalzgraf, Jan Schehl, Jochen Steigner, and Colette Weihrauch for the implementation and evaluation of the dialogue infrastructure.

References

[Allen et al., 2000] Allen, J., Byron, D., Dzikovska, M., Ferguson, G., Galescu, L., and Stent, A. (2000). An Architecture for a Generic Dialogue Shell. *Natural Language Engineering*, 6(3):1–16.

[Apt and van Emden, 1982] Apt, K. R. and van Emden, M. H. (1982). Contributions to the theory of logic programming. *J. ACM*, 29(3):841–862.

[Benjelloun et al., 2008] Benjelloun, O., Das Sarma, A., Halevy, A. Y., Theobald, M., and Widom, J. (2008). Databases with uncertainty and lineage. *The VLDB Journal*, 17(1):243–264.

[Bizer, 2009] Bizer, C. (2009). The emerging web of linked data. *IEEE Intelligent Systems*, 24(5):87–92.

[Buneman and Tan, 2007] Buneman, P. and Tan, W.-C. (2007). Provenance in databases. In *SIGMOD Conference*, pages 1171–1173, New York, NY, USA.

[Cimiano et al., 2007] Cimiano, P., Haase, P., and Heizmann, J. (2007). Porting natural language interfaces between domains: an experimental user study with the orakel system. In *IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces*, pages 180–189, New York, NY, USA. ACM.

[Fensel et al., 2003] Fensel, D., Hendler, J. A., Lieberman, H., and Wahlster, W., editors (2003). *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press.

[Glass et al., 2008] Glass, A., McGuinness, D. L., and Wolverton, M. (2008). Toward establishing trust in adaptive agents. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces*, pages 227–236, New York, NY, USA. ACM.

[Lopez et al., 2005] Lopez, V., Pasin, M., and Motta, E. (2005). Aqualog: An ontology-portable question answering system for the semantic web. In *Proceedings of ESWC*, pages 546–562.

[Martin et al., 1999] Martin, D., Cheyer, A., and Moran, D. (1999). The Open Agent Architecture: a framework for building distributed software systems. *Applied Artificial Intelligence*, 13(1/2):91–128.

[Ramachandran and Krishnamurthi, 2009] Ramachandran, V. A. and Krishnamurthi, I. (2009). Nlion: Natural language interface for querying ontologies. In *COMPUTE '09: Proceedings of the 2nd Bangalore Annual Compute Conference on 2nd Bangalore Annual Compute Conference*, pages 1–4, New York, NY, USA. ACM.

[Reithinger et al., 2005] Reithinger, N., Fedeler, D., Kumar, A., Lauer, C., Pecourt, E., and Romary, L. (2005). MIAMM - A Multimodal Dialogue System Using Haptics. In van Kuppevelt, J., Dybkjaer, L., and Bernsen, N. O., editors, *Advances in Natural Multimodal Dialogue Systems*. Springer.

[Schehl et al., 2008] Schehl, J., Pfalzgraf, A., Pfleger, N., and Steigner, J. (2008). The BabbleTunes System. Talk to Your IPod! In *Proceedings of the 10th International Conference on Multimodal Interfaces (ICMI)*.

[Seneff et al., 1999] Seneff, S., Lau, R., and Polifroni, J. (1999). Organization, Communication, and Control in the Galaxy-II Conversational System. In *Proceedings of Eurospeech'99*, pages 1271–1274, Budapest, Hungary.

[Sonntag, 2010] Sonntag, D. (2010). *Ontologies and Adaptivity in Dialogue for Question Answering*. AKA and IOS Press, Heidelberg.

[Sonntag et al., 2009] Sonntag, D., Deru, M., and Bergweiler, S. (2009). Design and Implementation of Combined Mobile and Touchscreen-Based Multimodal Web 3.0 Interfaces. In *Proceedings of the International Conference on Artificial Intelligence (ICAI)*.

[Sonntag et al., 2007] Sonntag, D., Engel, R., Herzog, G., Pfalzgraf, A., Pfleger, N., Romanelli, M., and Reithinger, N. (2007). *SmartWeb Handheld—Multimodal Interaction with Ontological Knowledge Bases and Semantic Web Services*, volume 4451 of *Lecture Notes in Computer Science*, pages 272–295. Springer.

[Sonntag and Kiesel, 2010] Sonntag, D. and Kiesel, M. (2010). Linked data integration for semantic dialogue and backend access. In *AAAI Spring Symposium on Linked Data Meets Artificial Intelligence*.

[Sonntag and Möller, 2009] Sonntag, D. and Möller, M. (2009). Unifying semantic annotation and querying in biomedical image repositories. In *Proceedings of International Conference on Knowledge Management and Information Sharing (KMIS)*.

[Suchanek et al., 2008] Suchanek, F. M., Kasneci, G., and Weikum, G. (2008). YAGO - A Large Ontology from Wikipedia and WordNet. *Elsevier Journal of Web Semantics*, 6(3):203–217.

[Theobald et al., 2010] Theobald, M., Sozio, M., Suchanek, F., and Nakashole, N. (2010). URDF: Efficient reasoning in uncertain RDF knowledge bases with soft and hard rules. Technical Report MPII20105-002, Max Planck Institute Informatics (MPI-INF), Saarbruecken, Germany.

[Wahlster, 2003] Wahlster, W. (2003). SmartKom: Symmetric Multimodality in an Adaptive and Reusable Dialogue Shell. In Krahl, R. and Günther, D., editors, *Proceedings of the Human Computer Interaction Status Conference 2003*, pages 47–62, Berlin, Germany. DLR.