Comprehension of RDF Data Using Situation Theory and Concept Maps

Jakub J. Moskal
VIStology, Inc.
Framingham, MA 01701, USA
Email: jmoskal@vistology.com

Mieczyslaw M. Kokar
Northeastern University
Boston, MA 02115, USA
Email: m.kokar@neu.edu

Brian E. Ulicny
VIStology, Inc.
Framingham, MA 01701, USA
Email: bulicny@vistology.com

***Abstract*—The amount of RDF data available on the Web has been increasingly growing over the past few years. Developing and fine-tuning SPARQL queries in order to sift through the data may be a very challenging task for human operators who need to quickly make sense of large graphs. In addition, often multiple queries need to be issued in order to gather and understand the context (relevant facts) for the explanation of the query. Thus, the challenge is not only to answer the query, but also to provide context, so that the analyst can easily comprehend what the data is actually conveying.**

This paper describes results of an investigation of the possibility to apply key aspects of Situation Theory, and its ontological realization in the Situation Theory Ontology, to simplify and abstract large RDF data sets, given a focus query from the analyst. In this approach, the query results are presented as concept maps. The approach was successfully implemented as a prototype, although this paper does not include a description of the tool.

I. Introduction

Development of intelligence products in various domains, e.g., business or military, requires sifting through tremendously large amounts of data, most of which so far is in an unstructured (or semi-structured) form (text reports, web pages). This constitutes a very high challenge to the analyst who performs this kind of task. While the analyst has in mind an idea of the focus of the inquiry, the focus may exist only in the analyst's head and thus cannot be supported by a computer-based tool. One way for the analyst to tell the computer what is being looked for is to issue a search query, e.g., using keywords. However, the tools that support keyword-based text search will return documents (or pointers to) that contain the words; the analyst still needs to do the hard work of reviewing the plethora of documents returned. Another way is to first use a text processing tool that will analyze the documents, extract entities and relations identified in those documents and represent them in a structured language, e.g., Resource Description Language (RDF) [1], and then analyze the resulting formal representation using an appropriate query language. An example of the development in this domain is the idea of *Linked Data* [2], which has resulted, among others, in a quite large knowledge base called *DBpedia* [3].

In fact, DBpedia is just one of the numerous open datasets that have been published in RDF format. As the chart in Figure 1 shows, the number of such datasets has been rapidly growing in the recent years. Unfortunately, the RDF structured

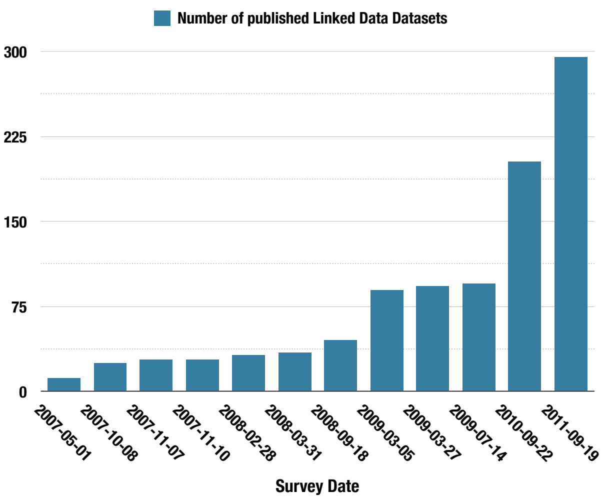

Fig. 1. Number of datasets that have been published in Linked Data format between 2007 and 2011 [4].

information is still very difficult to analyze. To illustrate the problem, consider an example of analyst query about a gang-related activity:

What were the circumstances of Richard H. Barter's death?

Such a query can be expressed in SPARQL query language [5] using the DESCRIBE query and the FILTER command that makes use of regex pattern matching to extract all the facts that are related to "Richard H. Barter". Even though DBpedia had only one resource ("Richard H. Barter") that is directly related to the query, the query returns more than 25 other resources that are one way or another related to this resource. DESCRIBE queries return RDF graphs and in order to analyze such an answer the analyst would have to go over all of the links and nodes and decide which of them are relevant.

Now the question is how to present the result of the query to the analyst? One of the formats for visual representation of complex information structures that has been proved quite successful in various uses, including knowledge structuring, learning and even knowledge creation, is the representation called *Concept Map* [6], [7], [8]. However, as discussed later in the paper, concept maps that are direct representations of

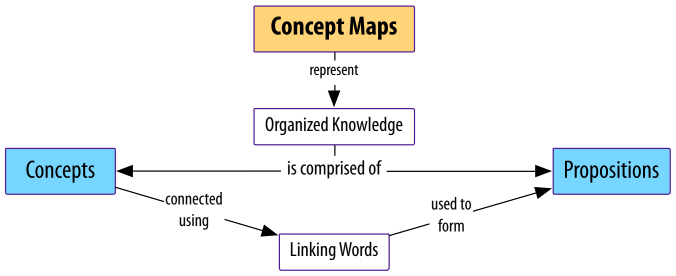

Fig. 2. Example of a concept map, representing the notion of concept maps itself [12].

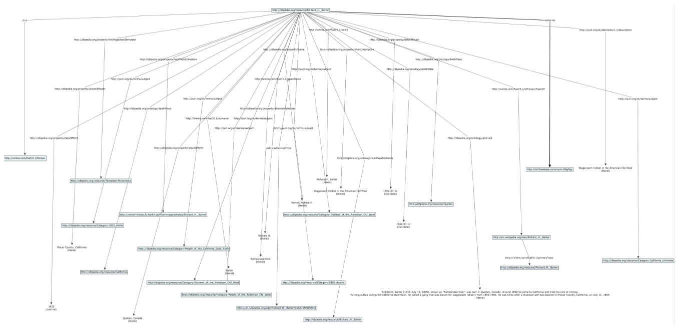

Fig. 3. Small RDF graph, returned by a SPARQL DESCRIBE query, visualized as a concept map using the IHMC CMap tool.

RDF graphs can also become quite complex and thus difficult to comprehend.

The problem addressed in this paper is the transformation of RDF to concept maps so that the resulting concept map is *relevant* to a specific analyst *query*, includes the appropriate *context*, and is presented in a more *abstract form* than the original RDF so that it is easy to comprehend. Our approach is to use key aspects of Situation Theory of Barwise and Perry [9], as extended and formalized by Devlin [10], map our problem to this theory and implement algorithms for constructing concept maps based on such a framework. In this work, we used the Situation Theory Ontology (STO) [11] that we developed earlier.

The rest of this paper is organized as follows. In Section II we briefly overview concept maps. In Section III we briefly discuss why Situation Theory is a good candidate for the solution. Then in Section IV we show how we can represent analyst queries in the STO ontology. This is followed by the discussion of domain inference in Section V and situation reasoning in Section VI. Section VII describes the derivation of (possibly) multiple contexts related to a query. Section VIII then discusses how the contexts are simplified in order to make the derived concept maps easier to comprehend. Finally, Section IX presents the conclusions of the paper and suggests some of the possible directions for future research.

II. Concept Maps

A *concept* is defined [8] as a perceived regularity or pattern designated by a label. *Propositions* are statements about some object or event in the universe, either naturally occurring or constructed. Propositions contain two or more concepts connected using linking words or phrases to form a meaningful statement. Sometimes these are called *semantic units*, or units of meaning.

Concept maps (c.f. Figure 2) include concepts (represented as boxes) and relationships between concepts (propositions) indicated by connecting lines linking pairs of concepts. Words in the boxes represent concept names, while words on/above the lines represent relationships between two concepts. Since concepts and properties are the building blocks of RDF, RDF graphs can be seen as concept maps. The CMap tools from IHMC can be used to provide graphical representations of RDF graphs as concept maps [8].

The concept map of Figure 2 shows five concepts and four propositions, where one of the concepts (Concept Maps) is a "meta-concept", since it represents the notion of a concept map itself. This map is only a fraction of a larger map, which shows the key features of concept maps [12]. Note that different look and feel styles can be applied to both concepts and linking words, e.g., different colors for different types of concepts.

Now returning to our example of the above described query, the CMap tool can load the answer to the SPARQL query and convert it to a concept map (c.f. Figure 3), however, the analysis of the map is still not that easy. One of the main reasons for this difficulty is the fact that concept maps generated in this way will contain too many concepts and relationships, many of them not relevant to the query. (Note: Clearly, Figure 3 is not readable. The sole purpose of this figure is to show the complexity of such concept maps.) One way to simplify the presentation would be to display just a small portion of the concepts and relationships. However, this operation needs to be performed very carefully so that important facts, without which the analyst would not be able to understand the answer to the query, are not omitted. Furthermore, the answer might include too detailed information, which clutters the global conceptual picture and defeats the purpose of the concept map. Hence, a fine balance between the simplicity and the amount of information must be kept in order to allow the analyst to quickly explore and understand the data.

III. Situation Theory

Situation Theory is "a set of mathematically-based tools to analyze, in particular, the way context facilitates and influences the rise and flow of information" [10]. Situation theory came about from the attempts to formalize *Situation Semantics* – reasoning about common sense and real world situations [9]. As postulated by Barwise and Perry, situations are first-class objects, i.e., they have their own existence, can stand in relation with other objects (including other situations) and can have their own attributes.

In situation theory, information about a situation is expressed in terms of *infons* written as:

$$\ll R, a_1, \ldots, a_n, 0/1 \gg$$

where R is an n-place relation and $a_1, \ldots, a_n$ are *objects* appropriate for R. Since situation theory is multi-sorted, the word "appropriate" means that the objects are of the types appropriate for a given relation. The last item in an infon is the *polarity* of the infon. Its value is either 1 (if the objects stand in the relation R) or 0 (if the objects don't stand in the relation R).

To capture the semantics of situations, situation theory provides a relation between situations and infons. This relationship is called the *supports* relationship which relates a situation with the infons that "are made factual" by the situation. Given an infon σ and situation s the *proposition* "s supports σ" is written as:

$$s \models \sigma.$$

The relation between a situation (in the world) and a representation of the situation (in a formal framework) is relative to a specific agent. It is the agent who establishes such a link. This link is defined by *connections* that link entities in the world to formal constructs of the situation-theoretic framework. These connections are not part of the formal theory. One refers to situations within a formal theory by using *abstract situations*, although the qualifier "abstract" is often dropped in most discussions of situation theory. An abstract situation is then a collection of infons supported by a specific situation.

In our approach we mapped key aspects of Situation Theory to Situation Theory Ontology (STO). The top-level classes of STO are shown in Figure 4. The details of this ontology were described elsewhere [13]. Here we just mention that the main idea behind this ontology is to capture the concept of "situation" (the Situation class serves this purpose). An individual s' of Situation that corresponds to a situation s in the real world, serves as the root to the description of the situation s. The abstract situation associated with s' is the *context*; it holds all the facts that are relevant to the situation, s. Other classes included in Figure 4 include Relation (to represent relations that individuals - instances of the class Individual - are involved in), Attribute (to represent attributes of both individuals and situations), Value and Dimensionality of the attributes, Rule (to represent rules for inferring higher-arity relations) and Polarity (to represent the values of Polarity; the only instances of this class are 1 and 0).

It is important to stress here that STO approximates Situation Theory by capturing the *supports* relation with a *entails* (or *derives*) relation, $\vdash$, between the collection of infons represening a situation and the infon representing a query [13]. Moreover, information is not represented in the form of infons. Instead, STO uses OWL and/or rules to represent knowledge about situations, i.e., abstract situations are captured by OWL sentences. However, as shown in Figure 4, STO includes the class ElementaryInfon. The sole role that ElementaryInfon plays in STO is to capture the focus of specific situations. I.e., queries (expressed in natural language) are formalized as instances of this class. ElementaryInfon resembles the structure of the infon in Situation Theory and thus has two

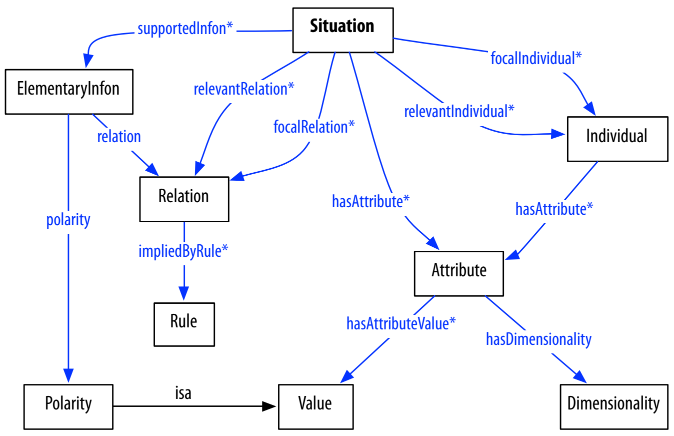

Fig. 4. Top-level classes in the STO ontology.

types of properties: *relation* (to point to the relation, R that the infon represents), and *anchor* (not shown in Figure 4) to point to the arguments of R. Polarity in STO is represented explicitly, i.e., positive assertions correspond to polarity 1 and negative assertions correspond to polarity 0.

One of the possible alternatives to Situation Theory that we looked at was the FrameNet approach [14]. FrameNet is based on a theory of meaning called "frame semantics" derived from the work of Fillmore et al. (cf. [15]). The basic idea is that the meanings of most words can best be understood on the basis of a semantic frame: a description of a type of event, relation, or entity and the participants in it.

While this idea seems to be close to that of the Situation Theory (ST) semantics, the latter has a number of advantages that make ST a better match for this particular problem. (1) ST grounds meaning in the world rather than in the language. This allows for the development of situation types that have meaning in the physical reality (e.g., battlefield, ships, missiles and so on), and not just in the syntax of the human language. (2) Unlike in more pure logic-based semantics, meaning in ST is provided by partial views of the world, not all possible worlds. This gives an advantage of being able to specify views of the world (situations) that are globally inconsistent, but locally consistent. This will allow analysts to specify situation types that, when taken together, are inconsistent. This capability allows the deference of the resolution of inconsistencies to the interactions with the world, rather than trying to develop a consistent set of types (an impossible state to achieve) before anything is utilized. (3) Situations in ST are first-class objects, i.e., they not only stand in relations with other situations, but can have their own attributes and properties. (4) In ST the meaning of a declarative sentence is a relation between utterances and described situations, which is exactly what is needed for a solution to our problem — developing concept maps that support the understanding of answers to specific analyst's questions (queries).

IV. Representing Queries

In our approach, the essence of the textual version of analyst queries needs to be extracted and mapped to to the ontology. Since situations are explicitly represented in STO, the mapping of the queries to STO has to be consistent with the intent of this ontology. In particular, since the intent is to connect a query with a context (which in STO is captured by a situation), as well as to ensure that the relevant facts are included in the context, queries were mapped to the class of ElementaryInfon and to a specific situation type.[1] For instance, the answer to the query whose textual representation is

"Did an insurgent visit a weapons cache"?"

can be captured by InsurgentWeaponsCacheSituation (a subclass of the Situation class), defined in OWL as follows:

InsurgentWeaponsCacheSituation ≡ Situation **and** (supportedInfon **some** (ElementaryInfon **and** (anchor1 **some** Insurgent) **and** (anchor2 **some** WeaponsCache) **and** (relation **value** visit)))

Answering such a query would involve inferring whether the current knowledge base supports the conclusion that there is a situation individual that is a member of the class InsurgentWeaponsCacheSituation. Note that the above definition assumes that the domain-specific ontology used in this query extends STO with some classes (e.g., Insurgent, WeaponsCache) and relations (e.g., visit).

Unfortunately, OWL is not sufficient enough to express some types of queries. For instance, the following query cannot be expressed in OWL alone:

"Which insurgents spied on a relative?"

The reason for this is that one needs to refer to variables, which are not supported by OWL. In particular, the intent of this query is to identify only those insurgents who spied on their own relatives, not just any insurgents who spied someone's relatives. In such cases one needs to use rules. For instance, using the STO, the query above could be expressed as the following rule:

Situation(s) $\wedge$ ElementaryInfon(i) $\wedge$ Object(a1) $\wedge$ Object(a2) $\wedge$ Relation(spiedOn) $\wedge$ supportedInfon(s,i) $\wedge$ anchor1(i, a1) $\wedge$ anchor2(i, a2) $\wedge$ relation(i, spiedOn) $\wedge$ Insurgent(a1) $\wedge$ Person(a2) $\wedge$ relative(anchor1, anchor2) $\rightarrow$ RelativeSpySituation(s)

Such rules can be captured in SPARQL 1.1 (using INSERT to assert new facts) or in an inference engine-specific language like BaseVISor's RDF-based BVR [16]. For the ease of use, since it was already the language in which some of the domain axioms were expressed (discussed below), BVR was chosen as the query language. In BVR, rules are defined within a rule base with each rule consisting of a *body* element and a *head* element (which can occur in either order). The name attribute can be used to assign a name to a rule base or rule. The heads and the bodies use the *triple syntax*, i.e., each rule consists of clauses, each being a triple (predicate, subject, object). The syntax of BVR is conceptually compatible with RDF. This kind of rules are easy to write and interpret; the only problem is that it is verbose. For this reason, BVR offers an abbreviated syntax [16].

[1]In OWL a query about some individuals can be viewed as a class, i.e., a collection of those individuals that satisfy the definition of the class.

The activities involved in the answering of analyst queries and creating concept maps that constitute the answers, is shown in Figure 5. The following sections describe each of these activities in more detail.

V. Domain Inference

The first step in the processing of an analyst query is to run the inference on the supplied RDF data and infer implicit facts about the domain (Step 1 in Figure 5). Since RDF does not provide strong axioms for inference, the RDF data can be augmented with additional axioms expressed in OWL and rules. OWL was the preferred choice, but if for some axioms it was not expressive enough, axioms were added in the form of BVR rules.

For instance, for the SynCOIN dataset [17] used in our experiments, examples of domain-specific axioms are definitions of object properties *associate* and *madeTransactionsWith*, both of which were defined as sub-properties of the transitive and symmetric *isConnectedTo* property (left side of Figure 6). An example of the use of these axioms is shown on the right side of the figure. Assuming that only John has been to a weapons cache, and that Mary is the only known insurgent, if the analyst issues a query *"Which known insurgents are connected to people who have been to a weapons cache?"*, the system should produce a map that includes Mary and John. In addition, the map should also include Bob and the relationships between all individuals, in order to fully represent the context. Without Bob in the result, it is not obvious how Mary and John are actually connected.

While the process of adding domain-specific axioms needs to be done manually, it is part of the knowledge engineering task, which is expected to be performed for each domain of application. Obviously, automatic ways of generating such axioms are desirable, but this was not part of this investigation. In our case, we arbitrarily decided which axioms to include.

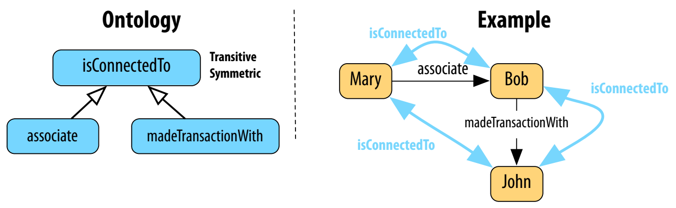

Fig. 6. Property taxonomy and example of its use. The blue lines represent implicit, inferred facts.

VI. Situation Reasoning

Once the domain inference is complete and all implicit domain facts are asserted in the knowledge base, individuals of a situation type that corresponds to the query, as well as relations among them, can be found (Step 2 in Figure 5).

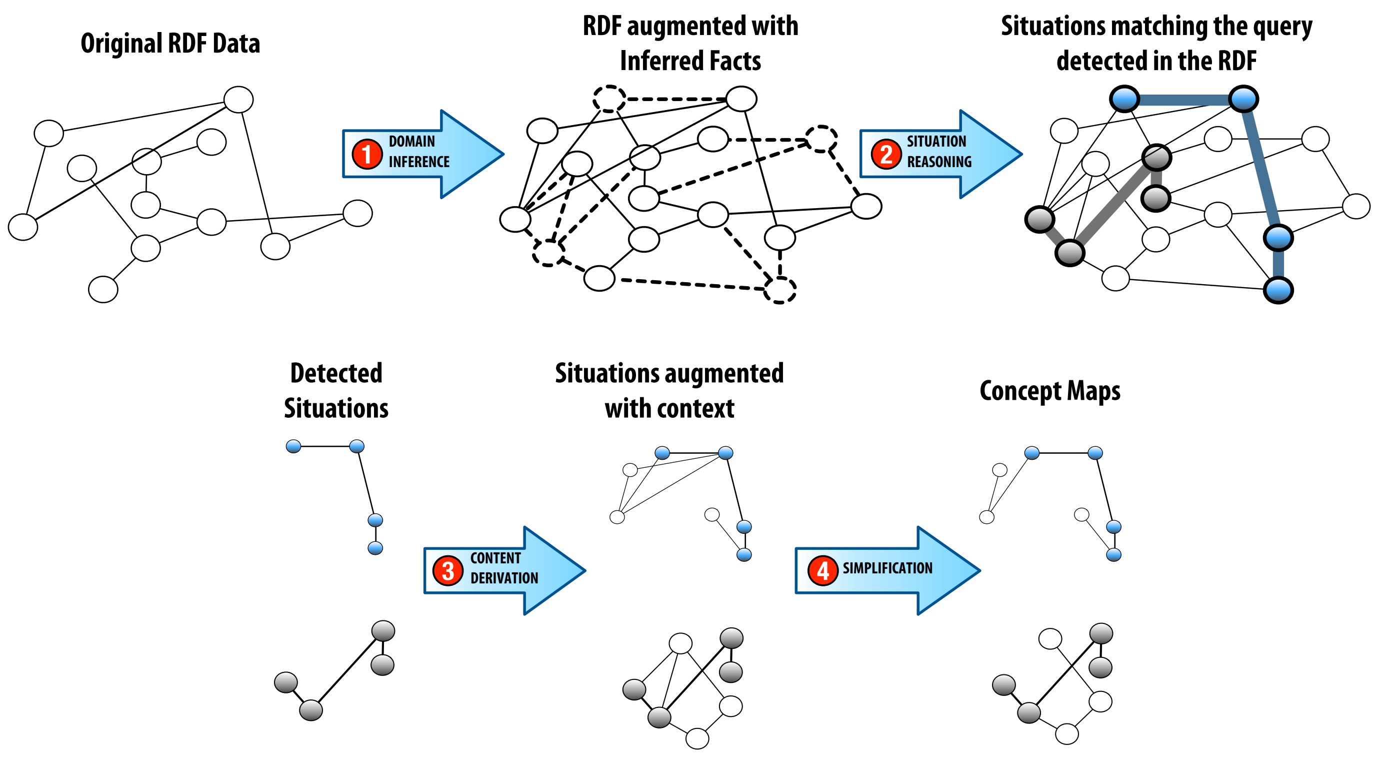
Original RDF Data
DOMAIN INFERENCE
RDF augmented with Inferred Facts
SITUATION REASONING
Situations matching the query detected in the RDF
Detected Situations
CONTENT DERIVATION
Situations augmented with context
SIMPLIFICATION
Concept Maps

Fig. 5. The process of transforming RDF data into comprehensible concept maps described in this paper. The situations detected in the RDF graph correspond to answers to an analyst query, which gives a focus for the produced concept maps.

To begin with, situation type definitions need to be analyzed — both those that are defined in pure OWL and those that are defined in rules (see Section IV for details). The main focus here is to extract the relations used in the definition of the situation types. For instance, for the InsurgentWeaponsCacheSituation type, the system should extract the *visit* relation. Similarly, for RelativeSpySituation, it would extract the *spiedOn* relation. Getting this information from OWL definitions is trivial, since we know the structure of the definition of situation types, which use the notion of ElementaryInfon, which in turn explicitly uses the object property *relation*. It gets more complicated with the situation types defined in rules. In our experiments, the BVR rule files were processed with regular expressions in order to find the relations. In the future, the rules themselves could be formally represented in OWL and the solution could avoid the use of regular expressions. Once the relations from situation type definitions are known, the process of asserting situation individuals is as follows:

- For each relation rel that is part of a situation type:
  - For each pair of individuals a_1 and a_2 that are associated with each other by the property rel:
    1) Assert that there is an individual s of RDF type sto:Situation
    2) Assert that there is an individual i of RDF type sto:ElementaryInfon, supported by situation s
    3) Assert the following facts: (i $anchor1$ a_1), (i $anchor2$ a_2) and (i $relation$ rel)

Now the reasoner can infer the situation types of the situation individuals.

VII. Derivation of Contexts

At this point, the answers consist of the anchors and the relations used in the situation definitions. For instance, for the weapons cache query and the axioms shown in Figure 6, given that John has been to a weapons cache, the system would return a basic concept map including Mary, John and isConnectedTo, but not include Bob and his relationships with them, which would explain why Mary is actually connected to John. Hence, the next processing step is to derive the context for the answer, i.e., find all individuals and relations that are relevant to the situation that represents the answer to the query. This corresponds to step 3 in Figure 5. Recall that "context" means an abstract situation, as described earlier in the paper. The main idea is that context is the description of a situation, including all the relevant individuals and the relevant relations among the individuals. All of this (the context) is captured by the *relevant facts*, i.e., facts that assert which individuals and relations are relevant and what are the relations among the relevant individuals.

For deriving context, we implemented a set of domain-independent rules, that backtrack some of the OWL inference rules. For instance, if a relation that is relevant to the query is defined as a property chain, the individuals and relations that form the chain are inferred to be relevant as well. Similarly, if a relevant relation is defined as a super-property of another property that holds between two relevant individuals, it is also inferred to be relevant. At the time of writing, the set of the context derivation rules is not complete, i.e., not every OWL inference rule that produces new facts has a corresponding relevance derivation rule. Also, some rules might produce facts that are not necessary to explain a situation to the analyst, thus producing some "noise". Such issues are on our agenda for future work.

As an example, the following describes one of the relevance derivation rules related to the transitive properties in OWL[2]:

- For a situation s, and a query q, if s satisfies the query:
  - For every fact (i_1 rel i_2) relevant to s and an individual i_3, if rel is a transitive property and if (i_1 rel i_3) and (i_3 rel i_2) are facts asserted in the knowledge base:
    1) Add (i_1 rel i_3) and (i_3 rel i_2) as facts relevant to s.

Figure 7 shows how derivation rules can be applied in the weapons cache example, given the axioms in Figure 6. First, based on the above rule applied to *isConnectedTo*, the inference engine would infer that the individual Bob is also relevant and should be part of the context (Figure 7b). Moreover, using a different derivation rule, the reasoner would infer that *associate* and *madeTransactionWith* are also relevant, because they are sub-properties of a relevant property and hold between relevant individuals (Figure 7c).

Note that not only individuals and properties are asserted as relevant to a situation, but entire facts (triples) are also asserted as such. It is not sufficient to just list the individuals and properties without showing the associations between them. In our experiments, we used the notion of OWL annotation properties in order to annotate facts as relevant to specific situation individuals. Since OWL does not support reasoning over annotation properties, the only way to implement such reasoning is to use rules. As we mentioned earlier in the paper, our preference was to use OWL reasoning first and add rules only out of necessity.

VIII. Simplification of Concept Maps

One can easily see that as a result of context derivation reasoning, the number of relevant facts for each situation might grow fast and if converted into a concept map, it could look quite convoluted (compare Figure 7a with Figure 7c). More importantly, it would most likely include redundant facts. For instance, Figure 7c shows that Mary and Bob are associated using two properties *isConnectedTo* and *associate*, although the former is just a generalization of the latter.

In order to make such resulting concept map less cluttered, and thus easier to comprehend, we need to remove facts that are relevant to a situation, but that are not necessary to comprehend the graph. We call this step *context simplification* and it corresponds to step 4 in Figure 5.

[2]Note that it is not important whether the facts on which the rule operates were derived or asserted by the user.

Similarly to the previous steps, for this purpose we developed a number of domain-independent rules that remove redundant facts. As an example, the following algorithm describes the rule that removes from a situation's context those properties whose sub-properties, holding between the same individuals, are relevant, yet not necessary:

- For a situation s, and a query q, if s satisfies the query:
  - For every relation r_1 and r_2 both relevant to s, if r_1 is a sub-property of r_2:
    * For every two facts (i_1 r_1 i_2) and (i_1 r_2 i_2) that are both relevant to s:
      1) Remove (i_1 r_2 i_2) from the context of s.

Back to the weapons cache example, based on the above rule applied to the graph in Figure 7, the system would remove the two *isConnected* links between Bob and the other two people, since they both provide redundant information. The *associate* and *madeTransactionWith* properties are more specific and clearly explain the context for the original query.

The resulting concept map could use different graphical styles when rendering concepts and links, in order to distinguish the query answer itself from its context. This approach gives the analyst a quick focus on the most important concepts in the graph, but also provides the context without cluttering the answer.

IX. Conclusion

The main objective of the research described in this paper was to investigate the possibility of using the ideas from Situation Theory (Barwise, Perry and Devlin), and its ontological realization in the Situation Theory Ontology, to the task of simplifying and abstracting concept maps, provided as RDF graphs, so that they are easier to comprehend by an analyst while still preserving the semantics of the original representation. This paper covers only some of the aspects of this investigation. In particular, it shows (by example) how an analyst's query can be mapped to an ontological representation, what it takes to derive facts that are relevant to the query, and how to represent such facts in graphical form (both with and without auxiliary facts that provide an explanation to the analyst of how they were derived). This investigation ended with a prototype tool (not included in this paper) for generating, displaying and manipulating concept maps in order to improve their comprehensibility. The next logical task for this research is to evaluate the tool on a representative number of queries and datasets and assess the approach with respect to its completeness and the strength of the rules used for the simplification of the query results. In particular, such an evaluation would require human-in-the-loop, i.e., the involvement of the analysts performing analyses of situational awareness in their domains.

Acknowledgment

This work was performed under Office of Naval Research contract N00014-14-P-1081 "Concept Maps from RDF (Resource Description Framework)". Any opinions, findings and

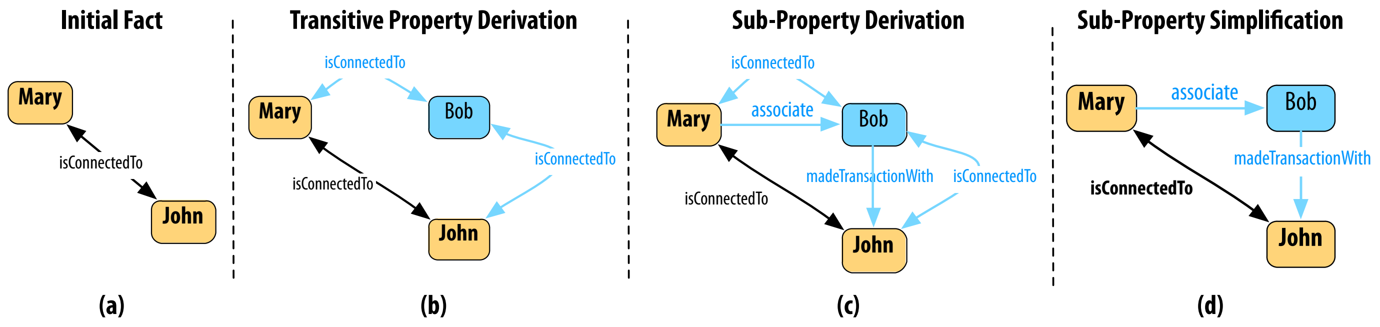

Fig. 7. Example of the context derivation and simplification of a query answer rendered as a concept map.

conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research. The authors would also like to thank the anonymous reviewers who provided many constructive suggestions for improving the presentation and for future research directions.

References

[1] W3C, "RDF semantics. W3C recommendation 10 february, 2004," Feburary 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

[2] T. Heath and C. Bizer, *Linked Data: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web: Theory and Technology*. Morgan & Claypool Publishers, 2011.

[3] DBPedia-Community, "DBpedia," September 2014. [Online]. Available: http://dbpedia.org/About/

[4] R. Cyganiak and A. Jentzsch, "The linked open data cloud diagram," http://lod-cloud.net/, September 19 2011.

[5] J. Pérez, M. Arenas, and C. Gutierrez, "Semantics and complexity of sparql," in *The Semantic Web-ISWC 2006*. Springer, 2006, pp. 30–43.

[6] J. D. Novak, "Concept maps and vee diagrams: Two metacognitive tools for science and mathematics education," *Instructional Science*, vol. 19, pp. 29–52, 1990.

[7] E. Plotnik, "Concept Mapping: A Graphical System for Understanding the Relationship between Concepts," *ERIC Clearinghouse on Information and Technology Syracuse NY*, vol. ED407938, 1997.

[8] J. D. Novak and A. Cañas, "The Theory Underlying Concept Maps and How to Construct and Use Them," cmap.ihmc.us/publications/researchpapers/theorycmaps/.

[9] J. Barwise and J. Perry, *Situations and Attitudes*. Cambridge, MA: MIT Press, 1983.

[10] K. Devlin, "Situation theory and situation semantics," in *Handbook of the History of Logic*, D. M. Gabbay and J. Woods, Eds. Elsevier, 2006.

[11] "Situation Theory Ontology." [Online]. Available: http://vistology.com/onts/2008/STO/STO.owl

[12] J. D. Novak and A. J. Cañas, "The theory underlying concept maps and how to construct and use them," Institute for Human and Machine Cognition, Tech. Rep. Technical Report IHMC CmapTools 2006-01 Rev 2008-01, 2008. [Online]. Available: http://cmap.ihmc.us/publications/researchpapers/theorycmaps/theoryunderlyingconceptmaps.htm

[13] M. M. Kokar, C. J. Matheus, and K. Baclawski, "Ontology-based situation awareness," *Information fusion*, vol. 10, no. 1, pp. 83–98, 2009.

[14] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet project," in *Proceedings of COLING/ACL*, 1998.

[15] C. J. Fillmore, "Frame semantics," in *Linguistics in the Morning Calm*. Seoul, Korea: Hanshin Publishing Co., 1982, pp. 111–137.

[16] C. J. Matheus, K. Baclawski, and M. M. Kokar, "BaseVISor: A triples-based inference engine outfitted to process RuleML and R-Entailment rules," in *Rules and Rule Markup Languages for the Semantic Web, Second International Conference on*, 2006, pp. 67–74.

[17] J. L. Graham, D. L. Hall, and J. Rimland, "A coin-inspired synthetic dataset for qualitative evaluation of hard and soft fusion systems," in *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*. IEEE, 2011, pp. 1–8.