Tracking RDF Graph Provenance using RDF Molecules *

Li Ding[1], Tim Finin[1], Yun Peng[1], Paulo Pinheiro da Silva[2], and Deborah L. McGuinness[2]

[1] Department of CSEE,
University of Maryland Baltimore County, Baltimore MD 21250
{dingli1,finin,ypeng}@cs.umbc.edu
[2] Knolwedge Systems Laboratory, Stanford University, Stanford CA 94305
{pp,dlm}@ksl.stanford.edu

Abstract. The Semantic Web can be viewed as one large "universal" RDF graph distributed across many Web pages. This is an impractical for many reasons, so we usually work with a decomposition into RDF documents, each of which corresponds to an individual Web page. While this is natural and appropriate for most tasks, it is still too coarse for some. For example, many RDF documents may redundantly contain the same data and some documents comprise large amounts of weakly-related or unrelated data. Decomposing a document into its RDF triples is usually too fine a decomposition, information may be lost if the graph contains blank nodes. We define an intermediate decomposition of an RDF graph G into a set of RDF "molecules", each of which is a connected sub-graph of the original. The decomposition is "lossless" in that the molecules can be recombined to yield G even if their blank nodes IDs are "standardized apart".
RDF molecules provide a useful granularity for tracking the provenance of or evidence for information found in an RDF graph. Doing so at the document level (e.g., find other documents with identical graphs) may find too few matches. Working at the triple level will just fail for any triples containing blank nodes. RDF molecules are the finest granularity at which we can do this without loss of information. We define the RDF molecule concept in more detail, describe an algorithm to decompose an RDF graph into its molecules, and show how these can be used to find evidence to support the original graph. The decomposition algorithm and the provenance application have both been prototyped in a simple Web-based demonstration.

1 Introduction

Eric Miller has characterized the Semantic Web as being a "web of data" rather than a "web of documents". Two of the features that account for this difference

* Partial support for this research was provided by DARPA contract F30602-00-0591 and by NSF awards NSF-ITR-IIS-0326460 and NSF-ITR-IDM-0219649.

are (i) that information is structured and encoded in a granularity much finer than the document level and (ii) that information is composed not out of words or other text elements, but using URIrefs denoting concepts and individuals.

One important and very useful attribute of RDF [1] is logical independence, i.e., one may freely combine RDF data found in different documents and coming from different locations into a unified graph. This raises two consequential issues: "How can RDF graphs be merged while preserving meaning?" and also "How can we decompose an RDF graph into constituent sub-graphs while maintaining meaning?". As a refinement of the second question, we can further investigate the granularity of a decomposition and ask "What are the smallest components into which an RDF graph can be decomposed without losing meaning?".

The "official" guideline for the *merge* operation on RDF graphs is provided by the semantics of RDF [2] and OWL [3]. In particular, OWL provides two features that are important to graph merging: the concept of *owl:FunctionalProperty* (FP) and *owl:InverseFunctionProperty* (IFP).

Berners-Lee and Connolly [4] defined an interesting *Diff* problem for RDF graph version control: how to implement the *merge* and *difference* operations on RDF graphs taking into account the functional dependencies between RDF nodes. This work analyzes graphs at the triple level of granularity and does not solve cases where a graph contains a blank node (BNode) that is not *functionally grounded*[3].

The triple level may be too fine a granularity on which to operate. For example, consider the graph in Figure 1 which describes a *foaf:Person* with first name "Li" and surname "Ding". If we decompose G1 into its two triples and treat each as a separate RDF graph, we lose the information that there exists a person that that both has the first name "Li" and also the surname "Ding". The graph has a single molecule containing both triples: (t1,t2).

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
{t1} (?x  foaf:firstName   "Li")
{t2} (?x  foaf:surname "Ding")
```

Fig. 1. The two triples in graph G1 assert that there is **a** foaf person who has a foaf firstName "Li" and a foaf surname "Ding". The only molecule (t1,t2) contains both triples.

In order to handle the information loss caused by triple level operation, we propose a higher level of granularity, the *RDF molecule*. An RDF graph's

[3] We will define functional grounding in the next section. Informally, a node in an RDF graph is functionally grounded if it is unique. Attaching an OWL inverse function property like *foaf:mbox* to a node makes it unique, and thus functionally grounds it.

molecules are the smallest components into which the graph can be decomposed into separate sub-graphs without loss of information.[4].

Consider the example shown in Figure 2. This graph asserts that the (unique) person who has mbox "dingli1@umbc.edu" also has a first name "Li" and a surname "Ding". The addition of the assertion about the *foaf:mbox* functionally grounds the blank node designated by ?x since this property is defined as an "inverse functional" property. The graph can be decomposed into two molecules, one with the mbox and firstname triples and another with the mbox and surname triples. The blank nodes in each molecule can be renamed, yet we are still able to combine the two molecules and re-construct the original graph.

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
{t1} (?x  foaf:firstName   "Li")
{t2} (?x  foaf:surname "Ding")
{t3} (?x  foaf:mbox   "dingli1@umbc.edu")
```

Fig. 2. The three triples graph G2 assert that **the unique** foaf person with foaf mbox "dingli1@umbc.edu" also has a foaf firstName "Li" and a foaf surname "Ding". There are two molecules: (t1,t3) and (t2,t3).

Finally, Figure 3 shows the graph with an additional blank node which represents a person with surname "Wang" who is the mother of the unique person with mbox dingli@umbc.edu. In this graph, the blank node identified by ?y is functionally ground by the combination of triples $t3$ and $t5$. The graph can be decomposed into three molecules: (t1,t3), (t2,t3), and (t3,t4,t5).

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix kin: <http://ebiquity.umbc.edu/ontologies/kin/0.3/>.
{t1} (?x  foaf:firstName   "Li")
{t2} (?x  foaf:surname "Ding")
{t3} (?x  foaf:mbox   "dingli1@umbc.edu")
{t4} (?y  foaf:surname "Wang")
{t5} (?y  kin:motherOf ?x)
```

Fig. 3. The four triples graph G3 assert that a foaf person with surname Wang is the mother of **the unique** foaf person with foaf mbox "dingli1@umbc.edu" also has a foaf firstName "Li" and a foaf surname "Ding". There are three molecules: (t1,t3), (t2,t3) and (t3,t4,t5).

One approach to decomposing an RDF graph into components is to use the concept of a *named graph* [5]. This allows one to circumscribe and name (using a

[4] As usual, we assume that when a graph is decomposed into sub-graphs the identifiers used for any blank nodes can be renamed

URIref) several sub-graphs within a single RDF document. Such named graphs are not necessarily minimal components since they can contain RDF sub-graphs of any size. Moreover, the publisher of an RDF graph has the responsibility of doing the decomposition into a set of named graphs. There is no automatic way of choosing the sub-graphs to name.

If all of the nodes in RDF graph are grounded, i.e. they are either URIrefs or Literals, an *RDF molecule* is essentially a triple. However, when a graph contains one or more blank nodes (BNodes), a *RDF molecule* may consist of varying number of triples since we do not want to break the link-semantic induced by BNodes.

We also identify some interesting application domains for *RDF molecule* as the following:

- **Tracking RDF graph provenance.** Provenance tracking is an important application for *RDF molecules*. Instead of finding the RDF document that contains a given RDF graph G, we may track G's provenance in finer granularity with decompose operation.
- **Evidence marshaling.** An RDF graph's molecules are the smallest meaning preserving subgraphs for which we might seek independent evidence and which can be easily combined to support the original graph.
- **RDF document version control.** A semantic *diff* operation for RDF graphs [4] enables one to describe changes to an RDF graph at the larger molecule level rather than at the level of triples. This is useful in tracking changes to a given ontology and building a patch file for different revisions.

Heuristic merging. Merging two RDF graphs is essentially taking the union of their triples subject to "standardizing apart" their blank nodes [2]. To reverse our decomposition, we make use of any inverse functional properties to further identify that two blank nodes necessarily refer to the same individual and subsequently merge them. In some applications, we might also use domain-specific heuristics that treat a set of properties as uniquely identifying a blank node. We could call this *heuristic grounding* to distinguish it from *functionally grounding*.

Such heuristics are common for many applications including natural language processing (e.g., in co-reference resolution), information extraction from text (e.g., named entity recognition) and mailing list management (e.g., identifying duplicates). There is a rich literature of approaches to this problem, ranging from work on databases [6] to recent work involving the semantic web [7]. Consider the example in Figure 4. Our heuristic might be that knowing either (i) a person's name and home phone number or (ii) a person's name and home address, is sufficient to uniquely identify a person.[5]. Using this heuristic, this graph has three molecules: (t1,t2,t3), (t1,t2,t4) and (t1,t3,t4).

Provenance at different granularities. Information on the semantic web can be viewed at different levels of granularity, from the universal graph formed from all of the RDF documents on the web, to individual documents and their

[5] This is a heuristic that will fail sometimes, as is the case of Heavyweight Boxer George Foreman and his sons

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
{t1) (?x  foaf:name   "Li Ding")
{t2} (?x  foaf:homePhone "410-555-1212")
{t3} (?x  foaf:homeAddress "1000 Hilltp Circle, Baltimore MD 21250")
{t4} (?x  foaf:age "27")
```

Fig. 4. Using a heuristic rule, we identify three molecules in this graph: (t1,t2,t3), (t1,t2,t4) and (t1,t3,t4).

parts. As Figure 5 shows, the RDF molecule is near the top of this hierarchy, just under the atomic triple. The data at each level will have associated justifications and provenance information as well. An RDF document can have its different parts annotated with provenance information, including named subgraphs, molecules and individual triples. If we extend the physics metaphor we could view individual URIrefs and literal values as sub-atomic particles. It may be of interest in some applications to study the provenance of these – for example, finding RDF documents that mention a particular URIref or use a certain literal string as a value.

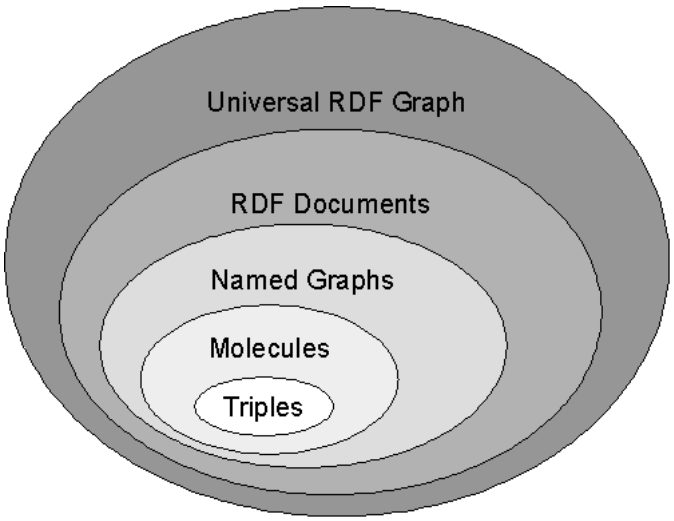


Fig. 5. Information on the semantic web can be viewed at different levels of granularity, from the universal graph formed from all of the RDF documents on the web, to individual documents and their parts. We introduce a new level, the RDF molecule, which is a connected sub-graph of a document.

Since the data are both justificands and comprise the justification, support can also be expressed at different levels of granularity. For example, we might

note that the justification for one triple comes from a particular set of triples or, if less precision is required, just identify the RDF documents in which the triples were found. In general, our approach allows a justificand, at any level of granularity, to have multiple justifications. Moreover, each of these can be rendered in multiple levels of granularity by moving up and down the data granularity hierarchy as required.

Contributions. This paper defines the concept of an *RDF molecule* and relates it to the notion of a lossless decomposition of an RDF graph. Using the RDF molecule concept, we develop an algorithm to fulfill lossless decomposition of RDF graphs. Finally, we demonstrate the utility of *RDF molecules* in tracking RDF graph provenance through a Web-based implementation.

Related Work. The *decompose* operation is very important in RDF graph modularization and has mainly been studied in the context of combining and partitioning large ontologies. Volz, Oberle and Maedche [8] have used an ontology-based approach by providing a set of terms to enrich the semantics of inter-ontology reference besides *owl:imports*. Stuckenschmidt and Klein [9] have used a statistical approach by analyzing the graph structure of large ontologies.

Most work on partitioning ontologies has treated it as a subjective issue, or at least one that requires some human judgment and decision making. There are seldom crisp criteria for grouping a set of classes and properties for a given topic. Our work is focused on objective criteria for decomposing RDF graphs into minimal components without information loss that can be automatically applied.

A recent automatic technique for partitioning ontologies is based on *"e-connections"* [10, 11]: an e-connection, which is a set of partitioned KBs, is generated from an input ontology O by iteratively analyzing the concept, roles and individuals in O from description logic perspective [12]. This feature is supported in the SWOOP ontology editor [13]. Our work does not limit in ontology partition problem, where a set of dependent description logic concepts are grouped together; in fact we are look for finest decomposition, which is suitable for tracking the provenance of an RDF graph.

Work on canonical RDF graphs is also related, although somewhat tangentially. This has been studied syntactically by Carroll [14] in the context of generating canonical representation of an RDF graph and logically (with RDF semantics inference support) by Gutierrez, Hurtado and Mendelzon [15] in an RDF database context.

2 RDF Graph Decomposition and RDF Molecule

This section gives formal definitions to RDF graph *decomposition* and *RDF molecue*, and then introduces two "lossless" implementations of RDF graph decomposition.

2.1 Basic Definitions

Definition 1. RDF graph decomposition
Given an RDF graph G and a background ontology W, a **decomposition** $\hat{G}$ *of G is a set of RDF graphs $G_1, G_2, ..., G_n$, where G_i is a subgraph of G. There are two operations related to decomposition:*

1. $\hat{G} = d(G, W)$ **decompose** *G to $\hat{G}$ using W;*
2. $G' = m(\hat{G}, W)$ **merge** *all elements (i.e. subgraphs) in $\hat{G}$ into a new RDF graph G' using W.*

Both operations are discussed in the context that no inferred triples will be produced during either operation.

Definition 2. *Given background ontology W, a pair of decompose/merge operations (d, m) is said* **lossless** *iff. given an RDF graph G, G is equivalent to $G' = m(d(G, W), W)$. We adopt* RDF graph equivalence *semantics in RDF [1].*

2.2 Labeling RDF Nodes

An RDF graph G has three disjoint sets of nodes, namely U, a set of URIrefs, L, a set of Literals, and B, a set of blank nodes or BNodes. The presence of blank nodes complicate RDF graph decomposition since BNodes do not come with universally unique identifiers. BNodes from different RDF graphs are assumed different by default. As mentioned by Berners-Lee and Connolly [4], some BNodes could be functionally grounded given a background ontology stating that some properties are instances of *owl:InverseFunctionalProperty* (IFP) or *owl:FunctionalProperty*(FP). We extend their definition to build a three-fold categorical partition on RDF nodes as the following:

- Given an RDF graph G, a node n in G is said **naturally grounded** (or grounded) if n is in either U or L.
- Given an RDF graph G with background ontology W, a node n in G is said **functionally grounded** on W if n is in B plus either of the following conditions is met:
  1. there exists a triple (n, p, o) in G, p is *IFP* according to W, and o is either grounded or functionally grounded.
  2. there exists a triple (s, p, n) in G, p is *FP* according to W, and s is grounded or functionally grounded.
- Given an RDF graph G with background ontology W, a node n in G is said **contextual grounded** if n is in B plus n is not *functionally grounded*.

A node n could be functionally grounded for different reasons, e.g. when both *foaf:homepage* and *foaf:mbox* are confirmed as *IFP* according to background ontology W, an instance of *foaf:Person* could functionally grounded on the persons homepage, and it could also be functionally grounded on the person's email.

2.3 Types of RDF Molecules

Definition 3. RDF molecule*(molecule).*
Given background ontology W, we may decompose an RDF graph G with a pair of lossless *decompose/merge operations (d, m). The elements of decomposition result are called* RDF molecules*. A* **RDF molecule** *m in G is a subgraph of G such that $m = d(m, W)$, i.e. m cannot be further decomposed.*

We further classify three basic types of RDF molecules:

Terminal Molecule (T-molecule). A *terminal molecule* only uses grounded nodes and/or functionally grounded BNodes, and all its BNodes are close. A BNode *bn* in a molecule n has two states, namely 'open' and 'close'. *bn* is said 'close' if it is functionally grounded and being used by exact one more triple in m, otherwise it is 'open'.

Non-Terminal Molecule (NT-molecule). A *non-terminal molecule* only uses grounded nodes and at least one functionally grounded BNodes, plus only one of its BNodes, i.e. the *active-functionally-grounded node*, is open. In fact, an *NT-molecule* can be better explained as the path that makes a *functionally grounded* node fgn (transitively) grounded on a grounded node gn in G. It is impossible to have two or more functionally grounded BNodes in open status in one molecule, otherwise it is decomposable.

Contextual Molecule (C-molecule). A *contextual molecule* uses at least one *context grounded* BNode(s). It is said *maximum* in an RDF graph G if it is not subgraph of any other *C-molecules* in G. In fact, maximum contextual molecules are the only possible *C-molecules* in lossless decomposition.

2.4 Naive Decomposition

We start with the simplest lossless decomposition without using any background ontologies, i.e. $W = \emptyset$. The corresponding implementation of *decompose* operation can be easily achieved: i) break a graph into a set of subgraphs each of which contains only one triple, ii) merge subgraphs who share the same BNodes until no more grouping can be done. Such decomposition produces only T-molecules and C-molecules. The decomposition can be demonstrate by Figure 6: the first result molecule (t1) is a T-molecule since both its subject and object are in U or L; the second result molecule (t2,t3,t4,t5) is a C-molecule since they share the same BNode $?x$.

2.5 Functional Decomposition

Now we can play with functional dependency among nodes. Background ontology may help us to find functionally grounded nodes and thus reduce the size of *C- molecules* derived in nave decomposition. The benefits of having functionally grounded nodes not only lie in labeling RDF triples with these functionally grounded nodes, but also help generate finer size RDF molecules. Hence we developed an implementation of "lossless" decompose operation $d_f(G, W)$ of RDF

```
{t1} (http://www.cs.umbc.edu/~dingli1  foaf:name "Li Ding")
{t2} (http://www.cs.umbc.edu/~dingli1  foaf:knows   ?x )
{t3} (?x  foaf:name  "Tim Finin")
{t4} (?x  foaf:mbox  "finin@umbc.edu")
{t5} (?x  foaf:mbox  "finin@cs.umbc.edu")
```

Fig. 6. The five triples graph G5 assert that a foaf person with foaf name "Tim Finin" and two mboxes "finin@umbc.edu" and "finin@cs.umbc.edu" is known by the foaf person with mbox "dingli1@umbc.edu" also has a foaf name"Li Ding".

graph $G(V, E)$ with background ontology W. The procedure is straightforward as the following:

1. Generate a molecule for each edge in G and classify it;
2. Generate all NT-molecules using functional dependencies derived from G and W;
3. Generate new T-molecules by combining two different NT-molecules sharing the same *active-functionally-grounded node*.
4. Generate new molecules by combining existing a C-molecule cm and an NT-molecule ntm when ntm's *active-functional-grounded node* $afgn$ is used by cm but not functionally grounded in cm, and then remove cm if ntm is a new C-molecule. Repeat this step until no new molecules are generated.
5. For each BNodes bn in G which are not used by any of the NT-molecules of G, generate a new molecule ncm by combining all C-molecules links to or from it, and then remove those C-molecules (since they all are subsets of ncm). At the end of iteration, all remainder C-molecules are *maximum C-molecules*.

The above operation $d_f(G, W)$ generates all possible molecules for G given background ontology W which specifies that *foaf:mbox* is an IFP. By applying it on the RDF graph in Figure 6, the result includes six T-molecules: (t1), (t2,t4), (t3,t4), (t2,t5), (t3,t5), and (t4,t5), plus two NT-molecules: (t4), (t5). Note that (t2,t3) is not recognized as T-molecule or NT-molecule since it has *contextual grounded* BNode $?x$, and it is neither recognized as C-molecule since $?x$ could be functionally grounded due to {t4}. Although the number of generated molecules may be much greater than the number of triples due to the molecule combination operation and they could be redundant to one another, they do enumerate all the smallest lossless information blocks of the original RDF graph.

3 Tracking RDF Graph Provenance using RDF Molecule

A useful feature of the Semantic Web is that users can use and reason over data distributed throughout the Web and created by many authors. Besides being guaranteed that inference procedure is trustworthy, users may also want to know the trustworthiness of the RDF graphs used as evidence/facts (the inference premise) in inference.

Since no one can guarantee that every RDF graph found on the Semantic Web is error free, provenance of the RDF graphs and their data is a good heuristic for evaluating trustworthiness. With provenance information, e.g. "where an RDF graph comes from" and "who has created an RDF graph", one may propagate his/her trust in an information source to trustworthiness judgments against each piece of data in RDF graph. Even when no information sources have been trusted *priori*, one could simply evaluate an RDF graph's trustworthiness by counting the number of sources asserting it. For example, we could believe in that "Tim Finin's email hash is XYZ" since it has been confirmed by more than seven RDF documents in the Web[6].

3.1 Building Semantic Web Provenance Service

Inference and provenance tracking are often two separate procedures: provenance information is rarely needed during inference since tracking provenance is often done before or after inference. Hence it is possible to separate provenance information from inference by providing a standalone provenance service. We can use conventional inference engines to process RDF graphs and leave the corresponding provenance information to one or several standalone provenance service provider(s). A basic operation of provenance service is to find a collection of RDF documents that directly assert a given RDF graph in whole or part. In order to build a semantic web provenance service, two design issues should be addressed here: (i) what kind of provenance information must be maintained and (ii) over what size or granularity should we seek provenance information.

For the first issue, we currently focus on document level provenance information since RDF documents are the standard way to make information encoded in RDF available.[7] Another possible granularity is *named graph*, but it is not yet popular since it requires syntactic and semantic extension of existing RDF specification. We are building an implementation that maintains provenance information at RDF document level: document provenance information, such as document URL, creator and inter-document dependency, is included in Swoogle's document metadata [16]. In addition, the provenance information for each triple is stored in 'quad' format (subject, predicate, object, source) in a MYSQL database without merging triples or generating inferred triples.

For the second issue, we focus on tracking provenance at the molecule level and triple level, i.e. the given RDF graph can be decomposed into small pieces which may be asserted by different sources. The two granularities serve different purposes. First, triple level provenance offers high *recall*, i.e. it finds all relevant information, even information that can "weaken" the given RDF graph. For example, an RDF graph $G1$ "(?x foaf:name "XYZZY") (?x foaf:mbox "a@for.com")" is relevant but does not help in justifying another RDF graph

[6] deciding to what degree these seven documents count as independent evidence is, of course, relevant and also a challenging problem.

[7] This may change as the semantic web evolves. RDF data can also be embedded in other objects such as XHTML documents, multimedia files and databases.

$G2$ "(?y foaf:name "ABC") (?y foaf:mbox a@foo.com)" even though $G1$ contains the second triple in $G2$. This situation exists because decomposing RDF graphs at triple level may not be *lossless*.

Second, molecule level provenance offers high *precision*, i.e. all the RDF documents asserting at least a molecule of the given RDF graph G do (partially) justify G. We may also note that the size of a complete list of molecules for an RDF graph could be very large due to combinational complexity[8].

While the triple level provenance service is straightforward, the molecule level provenance service is done as the following:

1. Given an RDF graph G with background ontology W, generate all possible molecules $M = \{m_1, m_2, ...\}$ using functional decompose operation $M = d(G, W)$.
2. For each RDF graph G_i in RDF database, check if any molecule in M is a subgraph of G_i.

3.2 Implementation and Evaluation

We have built a prototype system[9] based on Swoogle to demonstrate this idea. That prototype consists three parts: a provenance service that tracks provenance of a non-devisable RDF graph (i.e. all its triples should be asserted by one RDF document); a functional decomposition service that decomposes any RDF graph into molecules using background ontology; and a directory service which publishes the merged personal information collected from FOAF documents.

The decomposition algorithm is evaluated using RDF documents collected by Swoogle. For those RDF documents intended to be ontologies (e.g foaf, rss, dc ontologies), most comprise only T-molecules while a few also have some C-molecules. The existence of C-molecules is mainly due to the use of *owl:Restriction* and *owl:Union*. For example, the inference web ontology[10] contains 684 triples and decomposes into 349 T-molecules, each with only one triple, and 78 C-molecules with between four (e.g., for *owl:Restriction* on cardinality) and eleven triples (e.g., as caused by the use of an *owl:unionOf*). For those RDF documents intended to populate instance data, we have studied two collections of RDF documents, RSS and FOAF documents:

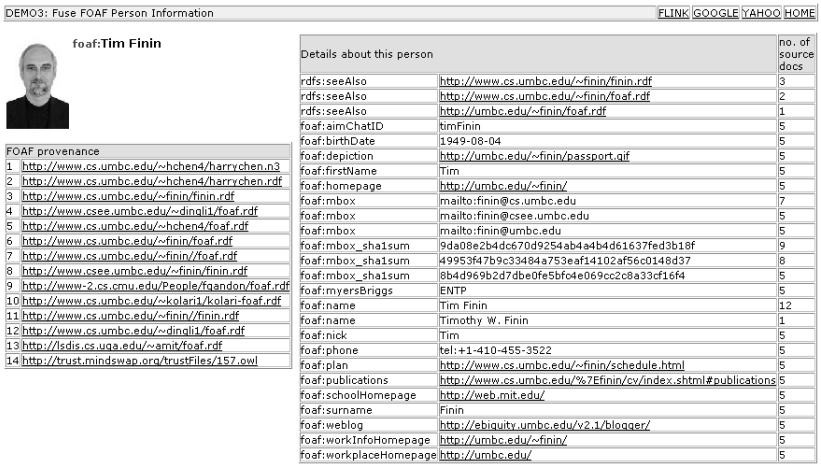
- RSS files have a regular decomposition pattern – many T-molecules and only one C-molecule. The C-molecule is usually an instance of *rss:items* that links to a *rdf:sequence* of *rss:item* instances.
- FOAF files have various decomposition patterns since the FOAF ontology takes advantage of inverse functional properties. Usually the number of generated molecules is less than the number of triples, but we have observed exceptions.

[8] A BNode could be functionally grounded according to many NT-molecules

[9] The service is available at http://swoogle.umbc.edu/service/provenance/ for experimentation.

[10] This ontology can be found at http://inferenceweb.stanford.edu/2004/07/iw.owl.

FOAF allows personal information about an individual person to be published in a completely distributed manner by many authors. It also also provides functional and inverse functional properties whose semantics enable the merging or fusing of information found in separate documents [17]. The person directory service essentially shows the result of merging personal profile from 4156 FOAF documents containing 32727 instances of *foaf:Person*. Figure 7 shows a merged profile for "Tim Finin" and it shows the source RDF documents and number of RDF documents that confirms each triple.



DEMO3: Fuse FOAF Person Information — FLINK | GOOGLE | YAHOO | HOME

foaf:**Tim Finin**

FOAF provenance	
1	http://www.cs.umbc.edu/~hchen4/harrychen.n3
2	http://www.cs.umbc.edu/~hchen4/harrychen.rdf
3	http://www.cs.umbc.edu/~finin/finin.rdf
4	http://www.csee.umbc.edu/~dinqli1/foaf.rdf
5	http://www.cs.umbc.edu/~hchen4/foaf.rdf
6	http://www.cs.umbc.edu/~finin/foaf.rdf
7	http://www.cs.umbc.edu/~finin//foaf.rdf
8	http://www.csee.umbc.edu/~finin/finin.rdf
9	http://www-2.cs.cmu.edu/People/fqandon/foaf.rdf
10	http://www.cs.umbc.edu/~kolari1/kolari-foaf.rdf
11	http://www.cs.umbc.edu/~finin//finin.rdf
12	http://www.cs.umbc.edu/~dinqli1/foaf.rdf
13	http://lsdis.cs.uga.edu/~amit/foaf.rdf
14	http://trust.mindswap.org/trustFiles/157.owl

Details about this person		no. of source docs
rdfs:seeAlso	http://www.cs.umbc.edu/~finin/finin.rdf	3
rdfs:seeAlso	http://www.cs.umbc.edu/~finin/foaf.rdf	2
rdfs:seeAlso	http://umbc.edu/~finin/foaf.rdf	1
foaf:aimChatID	timFinin	5
foaf:birthDate	1949-08-04	5
foaf:depiction	http://umbc.edu/~finin/passport.gif	5
foaf:firstName	Tim	5
foaf:homepage	http://umbc.edu/~finin/	5
foaf:mbox	mailto:finin@cs.umbc.edu	7
foaf:mbox	mailto:finin@csee.umbc.edu	5
foaf:mbox	mailto:finin@umbc.edu	5
foaf:mbox_sha1sum	9da08e2b4dc670d9254ab4a4b4d61637fed3b18f	9
foaf:mbox_sha1sum	49953f47b9c33484a753eaf14102af56c0148d37	8
foaf:mbox_sha1sum	8b4d969b2d7dbe0fe5bfc4e069cc2c8a33cf16f4	5
foaf:myersBriggs	ENTP	5
foaf:name	Tim Finin	12
foaf:name	Timothy W. Finin	1
foaf:nick	Tim	5
foaf:phone	tel:+1-410-455-3522	5
foaf:plan	http://www.cs.umbc.edu/~finin/schedule.html	5
foaf:publications	http://www.cs.umbc.edu/%7Efinin/cv/index.shtml#publications	5
foaf:schoolHomepage	http://web.mit.edu/	5
foaf:surname	Finin	5
foaf:weblog	http://ebiquity.umbc.edu/v2.1/blogger/	5
foaf:workInfoHomepage	http://umbc.edu/~finin/	5
foaf:workplaceHomepage	http://umbc.edu/	5

Fig. 7. Fusing Dr. Tim Finin's person information

By tracking triple level provenance, we may sometime encounter some suspicious cases calling for investigation to check out the source RDF documents. For our example, we might ask (i) is the provenance for triple (?x rdf:type foaf:Person) useful information?; (ii) From where did an unfamiliar triple come, e.g., '?x foaf:myersBriggs ENTP'; and (iii) how were the three different email hashes associated with this person? The molecular level provenance helps to tackle these questions since the triple (?x rdf:type foaf:Person) will never be found alone. It is also easy to answer the other two issues by tracking corresponding molecules' provenance.

4 Conclusion and Future Work

We have defined the concept of an RDF molecule as a minimal component in a lossless decomposition of an RDF graph. RDF molecules provide a granularity for Semantic Web information that lies between that of an RDF document and

of an RDF triple. We have demonstrated and implemented an automatic algorithm that takes an RDF graph and produces all of the RDF molecules in the graph. RDF molecules have several direct applications, including data provenance tracking, evidential marshaling, semantics-based graph comparison and managing inference tasks. A demonstration of molecular graph decomposition and it's use in data provenance tracking is available on the web.

Our future work will focus on three areas: expanding the notion of decomposition to include heuristic grounding, exploring the utility of molecular decomposition for web based provenance discovery and integrating the molecular view into Inference Web [18]. To support the heuristic merging of blank nodes we plan to develop a representation to define the heuristics. A simple use case is to define a boolean combination of properties as being heuristically inverse functions. More complicated cases may require the use of a Semantic Web compatible rule language like RuleML.

The second of these tasks requires extensions to the Swoogle RDF search engine [16] to efficiently retrieve documents containing given molecules. We have a working prototype of some of the extensions, but more work is required to make it effective at a web-scale.

The last task will involve adapting these ideas to support Inference Web's framework for explaining conclusions [19, 20] reached by a Semantic Web reasoner. In particular, we plan on using PML to annotate the provenance of a graph components at several levels of granularity – triple, molecule, named sub-graph and document.

References

1. Klyne, G., Carroll, J.J.: Resource description framework (rdf): Concepts and abstract syntax. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (2004)
2. Hayes, P.: Rdf semantics. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/ (2004)
3. Dean, M., Schreiber, G.: Owl web ontology language reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210/ (2004)
4. Berners-Lee, T., Connolly, D.: Delta: an ontology for the distribution of differences between rdf graphs. http://www.w3.org/DesignIssues/Diff (2004)
5. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. Technical Report HPL-2004-57, HP Lab (2004)
6. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association (1969)
7. Guha, R.V.: Object co-identification. In: Proceedings of the AAAI Spring Symposium on Semantic Web Services, AAAI Press (2004)
8. Volz, R., Oberle, D., Maedche, A.: Towards a modularized semantic web. In: Proceedings of the ECAI'02 Workshop on Ontologies and Semantic Interoperability. (2002)
9. Stuckenschmidt, H., Klein, M.C.A.: Structure-based partitioning of large concept hierarchies. In: International Semantic Web Conference. (2004) 289–303
10. Kutz, O., Lutz, C., F.Wolter, Zakharyaschev, M.: E-connections of abstract description systems. Artificial Intelligence **156** (2004) 1–73

11. Kutz, O.: E-connections and logics of distance. PhD thesis, University of Liverpool (2004)
12. Grau, B.C., Parsia, B., Sirin, E., Kalyanpur, A.: Automatic partitioning of owl ontologies using e-connections. Technical report, University of Maryland Institute for Advanced Computer Studies (UMIACS) (2005)
13. Kalyanpur, A., Parsia, B., Hendler, J.: A tool for working with web ontologies. In: In Proceedings of the International Journal on Semantic Web and Information Systems, Vol. 1, No. 1, Jan - March. (2005)
14. Carroll, J.J.: Signing rdf graphs. Technical Report HPL-2003-142, HP Lab (2003)
15. Gutierrez, C., Hurtado, C., Mendelzon, A.O.: Foundations of semantic web databases. In: PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA, ACM Press (2004) 95–106
16. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., , Sachs, J.: Swoogle: A search and metadata engine for the semantic web. In: Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management. (2004)
17. Ding, L., Zhou, L., Finin, T., Joshi, A.: How the Semantic Web is Being Used: An Analysis of FOAF. In: Proceedings of the 38th International Conference on System Sciences,Digital Documents Track (The Semantic Web: The Goal of Web Intelligence). (2005)
18. McGuinness, D.L., Pinheiro da Silva, P.: Explaining answers from the semantic web: The inference web approach. Journal of Web Semantics **1** (2004) 397–413
19. Pinheiro da Silva, P., McGuinness, D.L., McCool, R.: Knowledge provenance infrastructure. IEEE Data Engineering Bulletin **26** (2003) 26–32
20. Welty, C., Murdock, J.W., Pinheiro da Silva, P., McGuinness, D.L., Ferrucci, D., Fikes, R.: Tracking information extraction from intelligence documents. In: Proceedings of the 2005 International Conference on Intelligence Analysis (IA 2005). (2005)