Tracking RDF Graph Provenance
using RDF Molecules [*]
(Revision 2)

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

Abstract. The Semantic Web facilitates integrating partial knowledge and finding evidence for hypothesis from web knowledge sources. However, the appropriate level of granularity for tracking provenance of RDF graph remains in debate. RDF document is too coarse since it could contain irrelevant information. RDF triple will fail when two triples share the same blank node. Therefore, this paper investigates lossless decomposition of RDF graph and tracking the provenance of RDF graph using RDF molecule, which is the finest and lossless component of an RDF graph. A sub-graph is *lossless* if it can be used to restore the original graph without introducing new triples. A sub-graph is *finest* if it cannot be further decomposed into lossless sub-graphs. The lossless decomposition algorithms and RDF molecule have been formalized and implemented by a prototype RDF graph provenance service in Swoogle project.

1 Introduction

The Semantic Web is well known as a "web of data" instead of the "web of documents". Knowledge in the Semantic Web is structured and organized at a finer level of granularity than free-text document, and the vocabulary consists of not only literal words but also URI based universal identifiers. These features of RDF [1] facilitate information integration (i.e., merging graphs from difference web sources into a unified one) and hypothesis test (i.e., finding evidence from web sources for the given hypothesis RDF graph). For example, one can build a comprehensive profile for a person by integrating knowledge from her own FOAF [3] document and others' FOAF documents that mentioned her. Here, tracking provenance information is a critical issue since it directly helps evaluating the data quality of the integrated information or the hypothesis. For example, the

---

[3] http://www.foaf-project.org/

quality of each entry of a merge personal profile can be evaluated by the amount and the quality of supporting sources.

In order to track the provenance of a target (integrated or hypothesis) RDF graph, we need to find supportive RDF graphs from web sources and pick an appropriate level of granularity to capture the overlapping sub-graph between the target graph and each source graph. As shown in figure 1, there could be various levels of granularity for tracking provenance of RDF graph, ranging from the universal RDF graph formed from all of the RDF documents on the web, to individual documents and their parts. However, most of them have limitations.
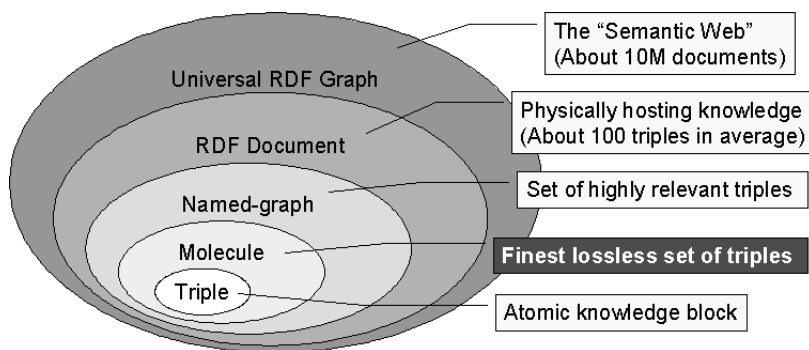

Fig. 1. The granularity of the Semantic Web ranges from the universal graph to triple, and the molecule is introduced in this paper.

First, RDF document and *Named graph* [2] are too coarse. Since the RDF graphs in them are fixed at creation time, they can contain both supportive triples for the target graph and additional irrelevant triples. Hence they are unable to offer a crisp bound for the overlapping sub-graph.

Second, RDF triple is too fine level of granularity for RDF graphs with blank nodes (BNodes) [4]. A blank node conveys both existential semantics and restriction semantics that binds triples using the same blank node; however, triple level comparison can cause information loss because it ignores the second semantics. For example, if we compare the three graphs in figure 2 at triple level, $G1$ overlaps $G2$ and $G3$ by having the triple that someone's surname is "Ding". However, $G3$ does not support $G1$ since they are describing different persons.

Therefore, we propose a new level of granularity – **RDF molecule**. Like the molecules in natural science, the molecules of an RDF graph are the finest components into which the graph can be decomposed without loss of information. [5] Molecules are not fixed at creation time; instead, they are the decomposition

---

[4] Blank node is used interchangeably with anonymous resource in this paper.

[5] 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
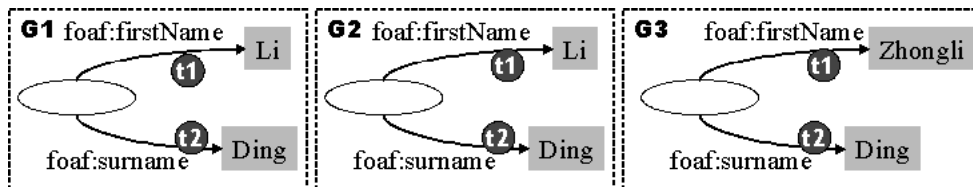
Fig. 2. The three RDF graphs above show personal information from three sources. The first one asserts that "a person who has a first name 'Li' and a surname 'Ding' ".

results of a given RDF graph. If an RDF graph is free of blank nodes, each triple constitutes a molecule; otherwise, triples are grouped into molecules according to the background ontology and the decomposition algorithm. In figure 2, each graph is a two-triple molecule.

Related Works. This work is influenced by works on canonical representation of RDF graphs, which seeks to compare two RDF graphs. Carroll's works on RDF graph matching [3] and canonical RDF graph [4] focus on syntactical level comparison in the context of matching and signing RDF graphs. Gutierrez, Hurtado and Mendelzon [5] focus on logical level comparison (which uses RDFS inference) in RDF database context.

There are also many works in partitioning large ontologies. Volz, Oberle and Maedche [6] developed an ontology for specifying inter-ontology reference besides *owl:imports*. *"e-connections"* [7, 8] offers a similar ontology framework but also provides an automatic partition algorithm from description logic perspective [9]. In contrast to these semantic approach, Stuckenschmidt and Klein [10] proposed a partition approach by analyzing the graph structure of large ontologies. These approaches concentrate on a higher level of granularity than instance, i.e., they decompose RDF graphs into bigger blocks, each of which clusters triples describing a set of semantically dependent classes and properties.

Our work aims at a different application domain – tracking provenance of RDF graph, i.e., decomposing the target graph and searching supportive RDF graphs. It works mainly in syntactic level with special treatment upon blank nodes and some terms in RDFS and OWL vocabulary; however, the main decomposition procedure does not require any RDFS or OWL inference. We only count in dependencies between triples caused by anonymous nodes but not semantic dependency between resources.

Contributions. This paper defines the concept of an *RDF molecule* and relates it to the notion of lossless decomposition of an RDF graph. Using this molecule concept, we developed algorithms 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.

---

the provenance of these – for example, finding RDF documents that mention a particular URIref or a literal string. However, this is out of scope of this paper.

2 RDF Graph Decomposition and RDF Molecule

This section defines *RDF graph decomposition* and *RDF molecule*, and then introduces two approaches to lossless RDF graph decomposition.

2.1 RDF Graph Decomposition

Definition 1. RDF graph decomposition.
An RDF graph decomposition *consists of three elements* (W, d, m)*: the background ontology* W*, the* **decompose** *operation* $d(G, W)$ *which breaks an RDF graph* G *into a set of sub-graphs* $\hat{G} = \{G_1, G_2, ..., G_n\}$ *using* W*, and the* **merge** *operation* $m(\hat{G}, W)$ *which combines all elements in* $\hat{G}$ *into the a unified RDF graph* G' *using* W*.*[6] *In addition, a decomposition must be* **lossless** *such that,*

$$\text{for any RDF graph } G,\ G = m(d(G, W), W).$$

When the elements in $\hat{G}$ *are disjoint,* $\hat{G}$ *is called an* **partition** *of* G*.*

In RDF graph decomposition, it is important to address node equivalence problem, i.e. whether two nodes in an RDF graph are equivalent or not. The "official" guideline is provided in RDF [11] and OWL [12] specifications: non-anonymous nodes in an RDF graph can be combined if they share the same URI; but merging anonymous nodes depends on the semantics provided by the background ontology. Berners-Lee and Connolly [13] mentioned the *functionally grounded* blank nodes, which can be compared using the semantics of *owl:InverseFunctionalProperty* (IFP) and *owl:FunctionalProperty*(FP). By running a little bit forward chain inference in the background ontology, such IFP and FP semantics can be further propagated via *owl:inverseOf* and *rdfs:subPropertyOf*. In addition, equivalence semantics such as *owl:sameAs*, *owl:equivalentClass* and *owl:equivalentProperty* can be used to compare related blank nodes as well.

Usually the nodes in an RDF graph G can be classified into three disjoint groups: U for RDF nodes with URIrefs, L for literals, and B for BNodes. Based on the above observations, we can classify RDF nodes into another three disjoint groups:

- A node n is **naturally grounded** (or grounded) if n is in either U or L.
- A node n is **functionally grounded** according to the background ontology W if n is in B and any 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.
  3. there exists a node n' in G such that n is equivalent to n' according to W, and n' is grounded or functionally grounded.

---

[6] Note that we restrict our discussion in the context where no inferred triples are produced, and we adopt *RDF graph equivalence* semantics from RDF [1].

– Given an RDF graph G with background ontology W, a node n in G is said **contextual grounded** if n is in B but n is not *functionally grounded*.

It is notable that a node n could be functionally grounded on multiple RDF nodes, e.g. when both *foaf:homepage* and *foaf:mbox* are confirmed as *IFP* according to background FOAF ontology, an instance of *foaf:Person* could functionally grounded on the homepage, the email, or both.

2.2 RDF Molecule

The simplest decomposition of a graph G is itself; however, users usually need the finest decomposition for tracking full spectrum of provenance information. Since triple level decomposition might cause information loss when RDF graphs contain blank nodes, we define the molecules of RDF graph.

Definition 2. RDF molecule*(molecule).*
RDF molecules *are the decomposition result of an RDF graph G. They are the finest and lossless sub-graph of G according to an decomposition (W, d, m).*

Based on the definition of molecule and the classification of RDF nodes in previous section, we identify three basic types of RDF molecules:

Terminal Molecule (T-molecule). A *terminal molecule* only contains grounded nodes and/or functionally grounded BNodes. All the BNodes in T-molecule are 'closed'. A BNode bn in a molecule m bn is called 'closed' if it is functionally grounded and being used by exactly two triples in m, otherwise it is 'open'.

Non-Terminal Molecule (NT-molecule). A *non-terminal molecule* only contains grounded nodes and at least one functionally grounded BNodes. Only one of the BNodes is 'open', and it is called the *active-functionally-grounded node*. [7] Intuitively, an *NT-molecule* is the path in RDF graph that makes a BNode (transitively) functionally grounded.

Contextual Molecule (C-molecule). A *contextual molecule* contains grounded nodes and at least one *contextual grounded* BNode(s). A C-molecule is *maximum* if it is not sub-graph of any other *C-molecules* in G. In fact, maximum contextual molecules are the only possible *C-molecules* in lossless decomposition.

The resulting RDF molecules are highly depended on the decompose operation and background ontology. For instance, NT-molecule will not be produced if the background ontology confirms no predicate in the given RDF graph is FP or IFP.

---

[7] Note that no two or more functionally grounded BNodes in 'open' state can co-exist in the same molecule.

2.3 Naive Decomposition

We start with the native decomposition without using background ontology. The corresponding *decompose* operation is essentially an algorithm that computes connected components. Note that only arcs connecting two blank nodes are of our interest according to the definition of molecules. Hence the complexity depends on the amount of such kind of arcs. A straightforward algorithm includes the following steps:

1. break a graph into a set of sub-graphs, each of which contains only one triple,
2. merge sub-graphs which share the same blank node until no more merge can be done.

Such decomposition produces only *T-molecules* and *C-molecules*. The decomposition can be demonstrate in figure 3: 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 $?x$. This decomposition is lossless since triples connected by BNodes are kept together.

```
{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. 3. The five-triple graph asserts 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" and a foaf name "Li Ding".

2.4 Functional Decomposition

While the nave approach is independent of background ontology, we can have a finer decomposition using the semantics of some special resources in OWL vocabulary, e.g. FP and IFP. With background ontology, we can figure out functionally grounded nodes and thus reduce the size of *C-molecules* by splitting triples grouped by functionally grounded BNodes. The corresponding decompose operation $d_f(G, W)$ is shown as the following:

1. Create a molecule for each triple in G and label the type of the molecule;
2. Generate NT-molecules using functional dependencies in G according to 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 BNode bn in G which is 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 sub-graph of ncm). At the end of iteration, all the residual C-molecules are *maximum C-molecules*.

The above operation $d_f(G, W)$ generates all possible molecules for G given background ontology W.

Consider the example shown in figure 4. 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 in the background ontology. The graph can be decomposed into two molecules, one with the mbox and name 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. A notable observation is that the molecule with name assertion could be found in much more web knowledge sources than the other molecule.

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

Fig. 4. The three-triple graph asserts that the unique foaf person with foaf mbox "dingli1@umbc.edu" also has a foaf name "Li Ding" and a foaf surname "Ding". There are two molecules: (t1,t3) and (t2,t3).

By applying a background ontology, which specifies that *foaf:mbox* is an IFP, over the RDF graph in figure 3, 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}. The number of generated molecules can be much greater than the number of triples because molecules are generated as a combinational result and they could be redundant to one another. However, molecules are smaller in size and do enumerate all finest possible information blocks conveyed by the original RDF graph. This feature is extremely important to exhaust all possible (partial) evidence for the original RDF graph.

Finally, figure 5 shows a more complicate situation where a graph contains two blank nodes. It is not decomposable using nave approach, but with functional decomposition, the blank node identified by ?y is functionally ground by the combination of triples $t3$ and $t5$. Hence, this 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. 5. The four-triple graph asserts 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".

2.5 Heuristic Merge and Decomposition

Merging two RDF graphs is essentially taking the union of their triples subject to "standardizing apart" their blank nodes [11]. 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. Intuitively, this is essentially the 'key' concept widely used database literature. 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 [14] to recent work involving the semantic web [15]. Consider the example in figure 6. 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.[8] Using this heuristic, this graph can be decomposed into three molecules: (t1,t2,t3), (t1,t2,t4) and (t1,t3,t4).

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

---

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

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 in the Web. Besides being guaranteed that the inference steps are reliable, users may also want to know the trustworthiness of the RDF graphs used as premises in inference, e.g. direct-assertion, evidence, and fact.

Since no one can guarantee that every RDF graph found on the Semantic Web is error free, provenance of RDF graph is an important hint for evaluating data quality. 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 knowledge source to each piece of knowledge in RDF graphs from the source. When no trust knowledge is available in *a priori*, one could simply evaluate an RDF graph's trustworthiness using vote. 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.[9]

3.1 Building Semantic Web Provenance Service

Inference and provenance tracking are often two separate procedures: provenance information is rarely needed during inference since provenance knowledge is usually needed 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) at what level of 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.[10] Another possible granularity is *named graph*, but it is not yet popular since it requires syntactic and semantic extension of current 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.

Regarding to 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

---

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

[10] 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.

pieces which may be asserted by different sources. The two levels of granularity 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 "john@foo.com")" is relevant but does not help in justifying another RDF graph $G2$ "(?y foaf:name "ABC") (?y foaf:mbox john@foo.com)" even though the second triple in $G1$ implies 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.[11] However, this situation is better than that no provenance information could be found.

While implementing triple level provenance service is straightforward, the Implementation of molecule level provenance service can be 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 $M = d_f(G, W)$.
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[12] based on Swoogle to demonstrate this idea. That prototype consists three parts:

– a provenance service that tracks provenance of a non-decomposable 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;
– 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[13] 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

---

[11] A BNode could be functionally grounded according to many NT-molecules
[12] The service is available at http://swoogle.umbc.edu/service/provenance/ for experimentation.
[13] This ontology can be found at http://inferenceweb.stanford.edu/2004/07/iw.owl.

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, and exceptions exist.

FOAF allows information about an individual person to be published in a completely distributed manner by many authors. It also provides inverse functional properties supporting information integration [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.
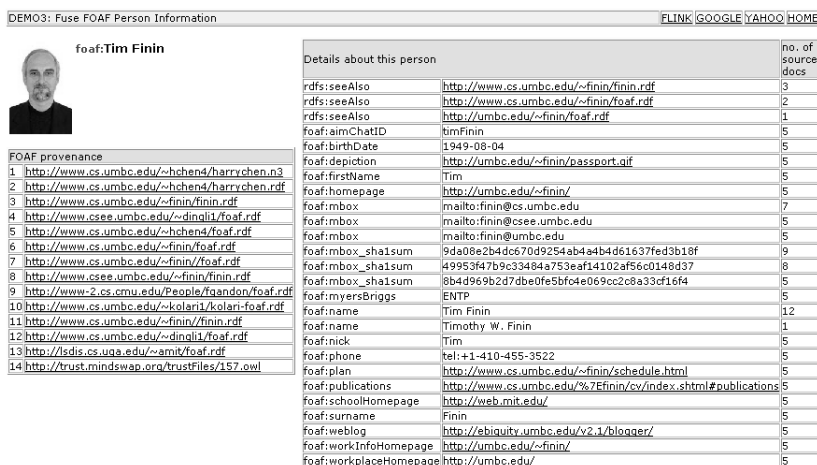

DEMO3: Fuse FOAF Person Information			FLINK GOOGLE YAHOO HOME
foaf:Tim Finin	Details about this person		no. of source docs
	rdfs:seeAlso	http://www.cs.umbc.edu/~finin/finin.rdf	3
	rdfs:seeAlso	http://www.cs.umbc.edu/~finin/foaf.rdf	2
	rdfs:seeAlso	http://umbc.edu/~finin/foaf.rdf	1
	foaf:aimChatID	timFinin	5
	foaf:birthDate	1949-08-04	5
FOAF provenance	foaf:depiction	http://umbc.edu/~finin/passport.gif	5
1 http://www.cs.umbc.edu/~hchen4/harrychen.n3	foaf:firstName	Tim	5
2 http://www.cs.umbc.edu/~hchen4/harrychen.rdf	foaf:homepage	http://umbc.edu/~finin/	5
3 http://www.cs.umbc.edu/~finin/finin.rdf	foaf:mbox	mailto:finin@cs.umbc.edu	7
4 http://www.csee.umbc.edu/~dingli1/foaf.rdf	foaf:mbox	mailto:finin@csee.umbc.edu	5
5 http://www.cs.umbc.edu/~hchen4/foaf.rdf	foaf:mbox	mailto:finin@umbc.edu	5
6 http://www.cs.umbc.edu/~finin/foaf.rdf	foaf:mbox_sha1sum	9da08e2b4dc670d9254ab4a4b4d61637fed3b18f	9
7 http://www.cs.umbc.edu/~finin//foaf.rdf	foaf:mbox_sha1sum	49953f47b9c33484a753eaf14102af56c0148d37	8
8 http://www.csee.umbc.edu/~finin/finin.rdf	foaf:mbox_sha1sum	8b4d969b2d7dbe0fe5bfc4e069cc2c8a33cf16f4	5
9 http://www-2.cs.cmu.edu/People/fgandon/foaf.rdf	foaf:myersBriggs	ENTP	5
10 http://www.cs.umbc.edu/~kolari1/kolari-foaf.rdf	foaf:name	Tim Finin	12
11 http://www.cs.umbc.edu/~finin//finin.rdf	foaf:name	Timothy W. Finin	1
12 http://www.cs.umbc.edu/~dingli1/foaf.rdf	foaf:nick	Tim	5
13 http://lsdis.cs.uga.edu/~amit/foaf.rdf	foaf:phone	tel:+1-410-455-3522	5
14 http://trust.mindswap.org/trustFiles/157.owl	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 "is it useful to track provenance for a triple like (?x rdf:type foaf:Person)?", "where did an unfamiliar triple come from, e.g., '?x foaf:myersBriggs ENTP"', and "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 RDF molecules as the finest components of a lossless decomposition of an RDF graph. RDF molecule provides a new level of granularity that lies between document and triple. We have demonstrated and implemented algorithms that generate molecules under difference context. Among many direct applications of RDF molecules, provenance tracking highlights the benefits through a web-based demonstration which tracks the provenance of RDF graph at molecule level. In addition, we also notice that RDF molecule can be used in the *Diff* problem of RDF graph as mentioned by Berners-Lee and Connolly [13]. For instance, a change of a triple with blank node will result in the removal of the molecules to which the triple belongs.

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 (or SWRL).

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 efficient.

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.

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. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. Technical Report HPL-2004-57, HP Lab (2004)
3. Carroll, J.J.: Matching RDF graphs. In: ISWC '02: Proceedings of the First International Semantic Web Conference on The Semantic Web, London, UK, Springer-Verlag (2002) 5–15
4. Carroll, J.J.: Signing RDF graphs. Technical Report HPL-2003-142, HP Lab (2003)
5. 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

6. Volz, R., Oberle, D., Maedche, A.: Towards a modularized semantic web. In: Proceedings of the ECAI'02 Workshop on Ontologies and Semantic Interoperability. (2002)
7. Kutz, O., Lutz, C., F.Wolter, Zakharyaschev, M.: E-connections of abstract description systems. Artificial Intelligence **156** (2004) 1–73
8. Kutz, O.: E-connections and logics of distance. PhD thesis, University of Liverpool (2004)
9. 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)
10. Stuckenschmidt, H., Klein, M.C.A.: Structure-based partitioning of large concept hierarchies. In: International Semantic Web Conference. (2004) 289–303
11. Hayes, P.: RDF semantics. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/ (2004)
12. Dean, M., Schreiber, G.: OWL web ontology language reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210/ (2004)
13. Berners-Lee, T., Connolly, D.: Delta: an ontology for the distribution of differences between rdf graphs. http://www.w3.org/DesignIssues/Diff (2004)
14. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association (1969)
15. Guha, R.V.: Object co-identification. In: Proceedings of the AAAI Spring Symposium on Semantic Web Services, AAAI Press (2004)
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)