Computing Information Value from RDF Graph Properties

Sinan al-Saffar  
Pacific Northwest National Laboratory  
Richland, WA 99354  
USA  
sinan.al-saffar@pnl.gov

Gregory Heileman  
University of New Mexico  
Albuquerque, NM 87131  
USA  
heileman@ece.unm.edu

ABSTRACT
Information value has been implicitly utilized and mostly non-subjectively computed in information retrieval (IR) systems. We explicitly define and compute the value of an information piece as a function of two parameters, the first is the potential semantic impact the target information can subjectively have on its recipient’s world-knowledge, and the second parameter is trust in the information source. We model these two parameters as properties of RDF graphs. Two graphs are constructed, a target graph representing the semantics of the target body of information and a context graph representing the context of the consumer of that information. We compute information value subjectively as a function of both potential change to the context graph (impact) and the overlap between the two graphs (trust). Graph change is computed as a graph edit distance measuring the dissimilarity between the context graph before and after the learning of the target graph. A particular application of this subjective information valuation is in the construction of a personalized ranking component in Web search engines. Based on our method, we construct a Web re-ranking system that personalizes the information experience for the information-consumer.

Categories and Subject Descriptors
H.1.1 [Information Systems]: Models and Principles-Systems and Information Theory[Value of Information]

Keywords
Semantic Web, Information Valuation, Semantic Search

1. INTRODUCTION
Information retrieval is a vital component of any information system, indeed the retrieval functionality is really the purpose of an information system. Retrieving information can be thought of as two-phased comprising searching and ranking. In searching several candidate solutions are discovered. For example all the documents indexed with a sought keyword are internally determined in the first phase of a Web search engine. In the second phase, all those candidate documents are examined and ranked before being returned in a prioritized list to the information consumer [7, 10]. The Semantic Web [9] extends the regular Web through enriching textual documents with Semantic markup that makes it easier for software agents to process the intended meaning of natural language. The Semantic Web can be viewed as a bottom up alternative to the earlier AI top-down attempts to extract meaning from natural language [21]. In the Semantic Web some of the burden is shifted to the information producer to clarify their intended meaning by semantically annotating their content which still happens with some automation from assisting software. The difference between AI and the Semantic Web is thus really that of how sophisticated the meaning-producing software needs to be. Regardless of the approach used to create the semantic model, the Semantic Web is comprised of documents whose meaning is processable by software agents. Machine-processable semantic annotations are typically expressed in families of logic-based languages that need to balance a trade-off between expressiveness for efficiency or even decidability. RDF [8] is one widely used such language comprised of binary predicates connecting two uniquely identifiable entities, a subject and an object. The connected entities and connecting predicate can be thought of as two vertices in a graph along with a connecting edge hence a set of these RDF binary predicates can be thought of as a directed graph, an RDF graph.

Web Ranking has thus far been carried out in a system and application-dependent manner whereas at its core, ranking is information valuation plus sorting. We need to formalize a definition and computation for the value of an information piece and simply employ such a formula in an information retrieval system be it a Web, Semantic Web, recommender system, or a closed-world knowledge base. Whether a system retrieves entire documents or answers specific questions is irrelevant to valuating information. These should simply be considered representational matters. Information value is subjective and should be calculated as a function of trust and impact on the information consumer’s context [2].

2. BACKGROUND AND RELATED WORK
While the searching stage of Web retrieval has employed term-based similarity methods, ranking has mostly relied on utilizing content relevance through measures such as term frequency-inverse document frequency (tf-idf) [7] and ap-
proximating global popularity through link analysis [17, 19]. Enhancements to the rank quality of Web information retrieval have been attempted through capturing more (in addition to the search keywords) of the consumer’s context such as preferences of certain subjects and bookmarked pages [22, 15]. Personalized rankings biased to such extended consumer contexts may naturally demand more resources but while this represents a challenge [16, 14], the usefulness of such limited optimization is questionable as they do not provide rank results that are much different than a global document rank [1].

Ranking in the Semantic Web has adapted techniques employed in the “regular” Web such as computing global popularity by analyzing link structure but including labeled RDF links in addition to regular anchor links [13, 20]. Moving beyond textual term matching in RDF meta data, researchers have performed semantic content analysis favoring concepts that are more central or more richly defined in an ontology or properties that are least likely to occur in a knowledge base [5, 4, 18] (these infrequent properties are assumed of provide higher informational content). Some of these methods require the information-consumer to explicitly specify a contextual parameter or two such as the unpredictability of the expected answer or the RDFS or OWL [12] classes to which the sought entities belong [4, 18]. Providing a consumer context is essential for making the ranking subjective and hence increasing information quality as we argued in [2]. An example of one way to map a consumer context using a special kind of graph was given in [3] where graphs were employed to compute a value for an information piece based on the trust in the provenance of the information and the causal impact it may have on it’s consumer’s context. We expand and refine those ideas in this work using standard RDF graphs only.

3. CREATION OF RDF TARGET AND CONTEXT GRAPHS

Given a target piece of information in natural language, a semantic RDF graph of the information of this document can be created either by the human author semantically annotating the document, or by a software agent or program. We created such a prototype as a Web browser plug-in. (more on this on the later sections). The target information piece is mapped to an RDF graph that represents the meaning of the document semantically. We denote this target RDF graph with $G_t$. Our goal is to determine the value of this graph subjectively (to a certain information consumer).

The information consumer’s preferences and world-view is represented with a set of assertions in RDF triples that also constitutes a graph. We call this the context graph, $G_c$. The context graph can be created through a software agent from monitoring and interacting with the information consumer and recording preferences. For example a simple context graph can be created from the semantic annotations from all the web pages in the consumers browser bookmark file or through a browser plug-in that extends its functionality with the ability to approve or disprove statements obtained from the semantic annotations on visited pages.

At the present time automated natural language processing abilities are somewhat limited but this is orthogonal to the fact that given the capability of the day to create semantic annotations from natural language, we should base information valuation on solid theoretic foundations assuming a certain mapping accuracy [2]. Information value is subjective. This is why we need a consumer’s context. This is not an implementation artifact. This is required in a theoretical definition of information value. The simplest consumer context is a query containing keywords. Such contexts are not differentiating enough to allow personalization of the results and understanding of the consumer needs. A more elaborate context is needed and this has been recognized in attempts to personalize search for example. Ideally all of a consumer’s context should be captured. This is difficult and may not be possible but we can assume that as much as possible, a world-view of the information consumer is captured in $G_c$, the more the technology enables us to make an accurate mapping, the better. But we submit that information value calculation depends on this context and can only improve as the context reflects reality better.

4. SEMANTIC INFORMATION VALUE

**Input:**

$G_t$: a target graph represent’s the information piece to be valuated.

$G_c$: a context graph represent’s the consumer’s context, or world-view.

**Output:**

$SIV(G_t, G_c) \in [0, 1]$, the Semantic Information Value of the content represented by $G_t$ to the information consumer with a context graph $G_c$. 

![Figure 1: Creation of target and context graphs.](image1)

![Figure 2: Learning as belief revision](image2)
4.1 Actual Semantic Information Value

**Definition 1.** The Actual Semantic Information Value of a target information piece, \( I_t \), is a function of two graphs and is denoted with \( SIV_a(G_t, G_c) \). The target graph, \( G_t \), is a semantic representation of \( I_t \) and the context graph, \( G_c \), is a semantic representation of the information consumer’s context:

\[
SIV_a(G_t, G_c) = \Delta G_c
\]

Where \( \Delta G_c \) is the amount of change \( G_c \) undergoes as a result of its consumer learning \( G_t \).

The actual value of \( G_t \) to a consumer may not be exactly predictable in the real world as the rules that govern the reasoning of the human consuming the information graph, \( G_t \), may not be capture-able or even knowable. However, in a closed-world knowledge system where inference rules for automated reasoning have been extracted from an expert, \( SIV_a \) may be computed by applying those inference rules to \( G_t \) to revise it into a new context graph, \( G_t^\prime \). Then we compute \( \Delta G_c \) by employing a suitable similarity (or more accurately dissimilarity in this case) to compare between \( G_c \) before the consumer learns \( G_t \) and \( G_c \) afterwards.

Let \( G_t, G_t^\prime \) denote the revised context graph or the context graph, \( G_c \), after the learning of \( G_t \) and let \( l \) denote some learning function representing the set of inference rules through which the information consumer may learn from assertions in \( G_t \) and generate new assertions to update \( G_c \). then:

\[
G_t^\prime = l(G_c, G_t)
\]

(2)

There are two cases where we can compute \( SIV_a \): Either we know \( l \) so we can obtain \( G_t^\prime \) by an automated procedure of reasoning, or we are given \( G_t^\prime \). In either case we can cast the information value of \( I_t \) as a graph dissimilarity problem:

\[
SIV_a(G_t, G_c) = \Delta G_c = disSimilarity_a(G_c, G_t)
\]

(3)

In the literature, the term similarity of two graphs is also used but these terms are really equivalent since:

\[
\text{dissimilarity}(G_1, G_2) = 1 - \text{similarity}(G_1, G_2)
\]

(4)

The dissimilarity of two graphs has been mostly computed based on either graph edit distance methods or finding the maximum common subgraph which have been shown to be equivalent [11]. For our domain we base the dissimilarity measure on edit methods where we assign costs to different graph modifying operations such as insertions, modification, and deletion and the dissimilarity between two graphs is based on the minimum cost incurred by the series of operations required to make the graphs isomorphic. Note the isomorphism in our case can be solved in linear time as the graphs are labeled so one combination needs to be tried for the matching as opposed to many \( (N!) \) in the general graph isomorphism problem.

We employ the following distance measure that gives uniform weight to operations thus degrading into using a count to compute the edit distance for calculating \( SIV_a \):

\[
disSimilarity_a(G_c, G_t) = \frac{|E_t | - |E_c |}{|E_t |}
\]

(5)

Where:

\[
disSimilarity_a(G_c, G_t) \in [0, 1]
\]

\(|E_c | = \) number of edges in the original context graph, \( G_c \), and,

\(|E_t | = \) number of edges in the revised context graph, \( G_t \) + the number of edges in \( G_c \) that got their labels changed.

We assume knowledge does not decrease so relationships between entities change but do not disappear. This is reflected by changes in the graph edge labels instead of deletions. We only use the number of edges as an indicator of graph growth as graphs grow by RDF triple units which is equal to an edge and two vertices so the edges suffice as an indicator. We do not consider it of different importance for a consumer to learn of a new relationship between two already known resources (vertices) or learn of an entirely new resource as we assume such importance would already (by definition) be reflected in the revision of the context graph.

We are using graph size, not structure as a measure of dissimilarity in the case of computing the actual semantic information value as the structure of the old context graph dictated this resulting size change (we do need to use structural properties of \( G_t \) when we want to predict \( G_t^\prime \) as we do in the next section). Sensitivity is uniform in computing the actual \( SIV \) where \( G_t^\prime \) is known and the context graph contains entities and properties whose importance is uniform. For example; even if some vertex is more connected than another then we do not assume that changed edges directed into such a vertex will have a higher impact on graph dissimilarity than changes in graph components near a less connected vertices. The reason behind this is that \( G_t^\prime \) is the graph resulting from applying all the inference to be made and thus all changes that did occur are already reflected in the revised graph, \( G_t^\prime \). We can not however make this same assumption when we consider sensitivity for \( SIV_a \) in the next section and we do not know \( G_t^\prime \) but predict what it could look like through such properties as connectedness. We considered \( SIV_a \) as a reference starting point to explain our idea of information value but in most systems we need to compute \( SIV_p \).

4.2 Predicted Semantic Information Value

In most systems including the Semantic Web where the actual rules governing the reasoning of the information consumer are not known, we can only compute \( SIV_p \), a prediction of the change in \( G_c \). As in the previous section \( SIV_p \) is cast as a graph dissimilarity problem except we adjust the sensitivity in assigning costs of the distance by adding weights to new edges according to the ideas of impact and trust. We compute \( SIV_p \) based on the potential impact \( G_t \) can have on \( G_c \) and the consumer’s trust in \( G_t \) inferred from degree of overlap (agreement) between the two graphs; \( G_c \) and \( G_t \).

We are given the target graph \( G_t \) representing the semantics of the target information to be evaluated, \( (I_t) \). This could be a single statement, a document, or a semantic association. We also know \( G_c \), the semantic RDF graph representing the consumer’s context before learning \( G_t \). We first compute \( G_t(l(G_t, G_c)) \), as a union of the two graphs. This represents all the possibilities the new knowledge is added to the existing one.

\[
G_t = G_t \cup G_c
\]

(6)

We use \( G_c \) and \( G_t \) to compute \( SIV_p \) similar to computing \( SIV_a \) in equation (3) but assign weights non-uniformly to the resulting \( G_t^\prime \) so higher weights get assigned to edges that may increase the size of \( G_c \) significantly through un-
known inferences. The structure of the context and target graphs and where they overlap gives us some idea of such possible revisions. One property we use is the degree of connectedness in the overlapping vertices. The intuition is that learning of new unknown (disconnected) concepts is not as important as learning related to already known concepts which in turn is less important than learning concepts that are centrally connected.

\[ SIV_{p} = \frac{\sum_{i=1}^{n} w(e_i) - \sum_{i=1}^{n} w(e_c)}{\sum_{i=1}^{n} w(e_i)} \quad (7) \]

One way to assign the weights to reflect the fact that connectedness is likely to result in higher graph growth due to inference can be:

\[ G_{t} = G_{t} \cup G_{c} \]

\[ w(e_{uv}) = t \times \text{degree}(v) \Rightarrow u \in G_{t}, v \in V_{t} \]

\[ w(e_{uv}) = t \times \text{degree}(u) \Rightarrow v \in G_{t}, u \in V_{t} \]

\[ w(e_{i}) = 1 \Rightarrow \text{otherwise} \]

\[ t = \text{Trust}(G_{t}, G_{c}) \text{ from equation}(8) \text{ and is explained next.} \]

The justification for the weight assignments in \( G_{t} \) above reflects impact and trust. We assign non-uniform costs to edges so that the edit distance will be higher for more valuable graph components. We base the sensitivity in assigning these costs on properties in the RDF graph that reflect the impact \( G_{t} \) can have on its consumer’s world-view, \( G_{c} \) and on the trust that consumer has in \( G_{t} \). We discuss these components:

### 4.2 Trust:

The graph, \( G_{t} \) changes \( G_{c} \) by an amount proportional to the amount of trust the information consumer has in \( G_{t} \). In our previous work we required the consumer to explicitly (maybe with the help of an assisting application) specify degrees of trust they have in the provenance of the information source of \( G_{t} \). In this work we employ the idea that trust can be inferred from the piece of information to be evaluated itself represented by \( G_{t} \). Such an approach is becoming more relevant with the proliferation of the implementation of linked data where additional meta data including provenance is becoming an inferable property and hence eventually becomes part of \( G_{t} \) by definition.

**Definition 2.** Trust is the degree of agreement between the target graph, \( G_{t} \) and the consumer’s context graph, \( G_{c} \).

We use the following equation to estimate this degree of agreement:

\[ \text{Trust}(G_{t}, G_{c}) = \frac{|E_{t} \cap E_{c}|}{|E_{t}|} \quad (8) \]

Where \( E_{t} \) are the edges in \( G_{t} \) and \( E_{c} \) are edges in \( G_{c} \). We normalize using \(|E_{t}|\) to account for the number of assertions made by the target document or information piece to be evaluated. A document stating all the facts in the world is surely to have some that agrees with our context. Thus trust is asymmetric for typically \( SIV_{t} \neq SIV_{c} \).

We are using the target information piece or document, \( I_{t} \) to be evaluated to deduce trust from the number of assertions already agreed upon between the document being evaluated and the consumer’s context. We divide the intersection by the size of \( G_{c} \) as \( G_{t} \) is expected to be smaller than \( G_{c} \) and narrower in domain. If an author made ten statements in a document and we agree with eight of them then our trust in this document is 80 percent. Trust is necessary but not sufficient in deciding the value of a target piece of information. We also need to measure impact.

### 4.2.2 Impact:

Even if an information piece represented by \( G_{t} \) is surely to be trusted, it is of little value if it is not relevant to its consumer’s context. If a consumer’s context is domain-focused on, say sculpting, a piece of information about basketball will be of less value to this consumer than a piece of information about an art gallery (the art gallery assertions are more likely to be connected with predicates to the sculpting components \( G_{c} \)).

If \( G_{t} \) had absolutely no overlapping vertices (entities) with \( G_{c} \), then \( G_{c} \) will be unchanged in becoming \( G_{t} \) except for the new disconnected component gained from equation (6). We assign the cost of 1 to the added edges of disconnected components in the new context graph, \( G_{c} \). In other words if a new RDF triple is added to an existing set of RDF triples and there is no overlap between the resources of the two sets, then the cost of the new RDF triple in computing the editing distance is equal to one. In such cases the information consumer learns of a new fact or set of facts about entities they didn’t even know existed before learning of \( G_{t} \). Such new information typically relates to domains of no interest to the information consumer. For example, a computer scientist being told that a new green butterfly is discovered in Africa (assuming \( G_{c} \) is accurate so no such concepts were in the context graph to begin with).

When a new RDF triple does overlap with a concept known to its consumer, as represented in \( G_{c} \), it may end up creating other assertions to be added to \( G_{c} \) through an unknown consumer reasoning process or set of rules. A computer scientist learning a new algorithmic bound doesn’t just gain that fact but his world-view, \( G_{c} \) get updated through some unknown-to-us set of rules that may add and modify other assertions in \( G_{c} \) to create \( G_{t} \). The more new knowledge is connected to existing knowledge the more valuable it is. We assign a cost value to new knowledge proportional to the degree of connectedness of the concepts (vertices) in \( G_{c} \) with which it overlaps. Learning concepts about entities that are highly connected in the original context graph, \( G_{c} \) is most valuable as this has high impact on the consumer’s world-view.

Impact is a function of connectedness of a vertex. If many vertices in \( G_{c} \) have directed edges into a certain vertex, \( v_{c} \) overlapping with a vertex, \( v_{t} \) in \( G_{t} \), this indicates a high likelihood of change in \( G_{c} \), as the document being evaluated may be discussing a central component in the consumer’s world-view.

\[ \text{Impact}(G_{t}, G_{c}) = \sum_{i=1}^{n} d_{i} \quad (9) \]

Where:

\[ d_{i} = \text{degree of } v_{c} \text{ in } G_{c} \text{ and } v_{c} \in V_{c} \cap V_{t} \]

\[ d_{i} = \text{degree of } v_{t} \text{ in } G_{c} \text{ and } v_{t} \in V_{t} \]

We have:

\[ SIV_{p} = f(\text{Impact}(G_{t}, G_{c}), \text{Trust}(G_{t}, G_{c})) \quad (10) \]

where \( f \) may be different functions to combine trust and impact in different ways. We use multiplication. In equation (7) we computed this \( SIV_{p} \) based on a particular weights.
assignment that models trust and impact.

Scalability: Since the similarity computation will need to be carried out frequently and the context graph representing the consumer’s preferences and world-view gets more useful as it grows in size, a scalable similarity function is critical. We only consider algorithms with an asymptotically linear time complexity in relation to the size of \( G_c \). The summations and set operations above are linear in the size of the input graphs.

When a consumer is presented with \( G_c \), and he believes all of its content, then maybe we are tempted to state that the new context is simply a \( G_{\text{f}} = G_1 \cup G_c \) not biased by new weights. This is not correct. The reason being is that the RDF graphs only deal with assertions in the knowledge base at question. What is missing are the rules. We do not know how a single assertion from \( G_1 \) will affect \( G_c \). It may not be simply added to \( G_{\text{f}} \) even if it were fully believed as there may be for example an inference rule that causes the addition of several new assertions or modifications of existing ones. These rules, unlike the case when calculating the actual \( SIV \) are, as we discussed, not known and may not be knowable. They may even depend on the mood of the consumer. However we can still estimate the potential impact \( G_c \) can have on \( G_1 \) by assigning more connected vertices higher weights as we did in calculating impact above. The reasoning is that vertices with more connections are likely to propagate any affect on them to the connected objects through the unknown reasoning process. We started with the union operation of \( G_1 \) and \( G_c \) to construct \( G_{\text{f}} \) but made the sensitivity of the dissimilarity nonuniform and dependent on the connectivity of the vertices by the way we assigned weights to the new components in \( G_{\text{f}} \). Added graph components on vertices that are more connected increase dissimilarity more than the same components intersect with a less connected area of \( G_c \). We further multiply this calculation by \( Trust(G_1,G_c) \) so a \( G_{\text{f}} \) that would have been very dissimilar to the original \( G_1 \) due to the many added graph components would be less dissimilar because there is little trust in the assertions as represented by the components being added. We calculate trust as an overlap which can be thought of as a similarity of the two graphs, \( G_1 \) and \( G_c \) hence there is a sweet point or trade-off between similarity (trust) and dissimilarity (impact) to contribute to the calculation of information value but overall after we predict \( G_{\text{f}} \), we are using its dissimilarity with \( G_c \) as the final measure of information value.

5. APPLICATION

We built a system to re-rank Web pages based on their predicted semantic information value, \( SIV_p \). We describe our experiments in this section.

5.1 Front-End Module

The front-end is built as a Web Browser Plug-in: This is a software add-on that extends the user interface and functionality of a Web browser. Our plug-in works with Mozilla Firefox. The plug-in is simply a front-end that sends the URL of the Web page presently displayed in the browser to our back-end server for semantic analysis. The plug-in also has a “semantic bookmark” button to allow the user to add the current page to their context graph, \( G_c \) which is maintained along with other user account details on the back-end server. The plug-in also has a log-in button as the system cannot function without identifying the consumer’s context graph to be used. The plug-in is implemented with AJAX and XUL. Figure 4 is a screen shot of our plug-in shown on the Firefox browser on Linux. After installing the plugin it can be used when you login by clicking the “Biased for button” which displays a window for logging in or registration. Hitting the “Evaluate” button will cause the current page to get analyzed on the back-end server. The result is a semantic value for the page and an optional explanation in the sidebar to the left.

The system enhances current browsing and search experience by providing a semantic value for the page being browse and displaying it in the browser plug-in UI on the tool bar. The plug-in also displays the information value on each link inside the browsed Web page. This serves as a guide for navigation so the consumer can pick the highest rank link if they wish. An explain button opens a sidebar to explain the calculation of the semantic value of the along with semantically identified entities (presently these are entities we could discover in dbpedia).

5.2 Back-End Module

The back-end consists or our application server which we wrote as a Java servlet which gets called from an Apache Tomcat server. Upon the server completing the semantic value computation of the page, it sends it back to the plug-in for display. The message sent back contains three parts, a semantic value for the present page, a semantic value for every page linked to from the present page, and an explanation of the semantic value which is a report that describes the identified entities on the page, \( G_1 \) along with their RDF class types ad the ones that intersect with the context graph. The servlet receives URL’s from front-end plug-in, downloads the text of the page, and creates the target RDF graph, \( G_1 \) by scanning for terms appearing as literal objects in the rdf:label property for a locally running dbpedia database[6]. We loaded dbpedia in Virtuoso RDF store. We have extracted the labels string URI pairs in a MySQL table for speeding up the lookup process. The back-end server also receives bookmark requests from the plug-in in which case the browsed page is semantically marked but added to the user’s context, \( G_c \). The back-end maintains users’ accounts and login passwords as well.

5.3 RDF Store and Database

We installed a full version of dbpedia, the semantically annotated Wikipedia on a dual core server running 64-bit Ubuntu Linux on eight Gigs of RAM. We used MySQL database to cache the text labels to URI mappings, an optimization to speedup the URI lookups. Installing Virtuoso and loading it with dbpedia was not a trivial experience especially that the dbpedia dataset contains more than 100 million triples! The implementation of the system presented many technical challenges. The main shortcoming is on the natural language processing front: we are able to map all the concepts from text to their respective URI’s but we usually have difficulty in mapping relationships which end up being the edges in the context graphs. We also implemented a version that uses a Reuters Web service that extracts URI’s from text but that too is not very good at extracting properties (a whole RDF triple from text where both the object and subject are in the text itself in addition to the prop-
7. REFERENCES


Example

Listing 1: Text (source of the triples in listing 2)

A good study of ethics can be found in "Beyond Good and Evil" by Frederick Nietzsche who influenced many philosophers and writers including Jean-Paul Sartre. Sartre discusses the human condition, ethics, and metaphysics in his novels and short
Listing 2: RDF Triples of target graph $G_t$ in fig. 3

```xml
<rdf:RDF>
  <rdf:Description rdf:about="d1:Jean-Paul_Sartre">
    <d0:mainInterests rdf:resource="d1:Metaphysics"/>
    <d0:mainInterests rdf:resource="d1:Ethics"/>
    <d0:schoolTradition rdf:resource="d1:Existentialism"/>
    <d0:deathplace rdf:resource="d1:France"/>
 </rdf:Description>

  <rdf:Description rdf:about="d1:Friedrich_Nietzsche">
    <d2:subject rdf:resource="d1:Category:Continental_philosophers"/>
    <d0:influenced rdf:resource="d1:Jean-Paul_Sartre"/>
    <d0:schoolTradition rdf:resource="d1:Existentialism"/>
   </rdf:Description>

  <rdf:Description rdf:about="d1:Beyond_Good_and_Evil">
    <d3:subject rdf:resource="d1:Ethics"/>
    <d3:author rdf:resource="d1:Friedrich_Nietzsche"/>
   </rdf:Description>

  <rdf:Description rdf:about="d1:Troubled_Sleep">
    <d3:author rdf:resource="d1:Jean-Paul_Sartre"/>
   </rdf:Description>
</rdf:RDF>
```

Figure 3: The target RDF graph, $G_t$, representing the semantics of listing 1

Figure 4: A screen shot of our semantic plugin
Figure 5: The context graph, $G_{c_3}$ before the consumer learning of $G_t$.

Figure 6: A magnified portion of $G_{c_3}$, which is $G_c$ (fig. 5) after learning of $G_t$ (fig. 3).