Journal of Universal Computer Science, vol. 13, no. 12 (2007), 1854-1880
submitted: 15/4/07, accepted: 31/7/07, appeared: 1/12/07 © *J.UCS*

On Ranking RDF Schema Elements (and its Application in Visualization)

Yannis Tzitzikas, Dimitris Kotzinos, Yannis Theoharis
(Computer Science Department, University of Crete, Greece, and
Institute of Computer Science, FORTH-ICS, Greece
{tzitzik, kotzino, theohari}@ics.forth.gr)

Abstract: Ranking is a ubiquitous requirement whenever we confront a large collection of atomic or interrelated artifacts. This paper elaborates on this issue for the case of RDF schemas. Specifically, several metrics for evaluating automatic methods for ranking schema elements are proposed and discussed. Subsequently the creation of a test collection for evaluating such methods is described, upon which several ranking methods (from simple to more sophisticated) for RDF schemas are evaluated. This formal way for evaluating ranking methods, apart from yielding credible and repeatable results, gave us some interesting insights to the problem. Finally, our experiences from exploiting these ranking methods for visualizing RDF schemas, specifically for deriving and visualizing top-k schema subgraphs, are reported.
Key Words: schema ranking, schema visualization, semantic web
Category: H.0, H.5

1 Introduction

As the modern society and economy increasingly depends on a deluge of digital information, the need for *ranking* becomes more and more crucial. The Semantic Web, has so far focused only on the task of ranking ontologies, a task that could aid the human-enacted process of ontology selection [Ding et al. 2005], [Patel et al. 2003], [Buitelaar et al. 2004], [Alani and Brewster 2005], [Sabou et al. 2006]. This paper focuses on the problem of ranking the elements of a single ontology; a problem that has not been studied in the literature so far. The motivation for this kind of ranking, is that it can alleviate the human effort required for understanding the contents and the structure of one ontology. This task is quite hard for ordinary users, also aggravated by the lack of satisfying ontology visualization tools, hence it is an obstacle for the realization and deployment of the Semantic Web. In addition, ranking could be exploited in a plethora of other tasks, e.g. for ordering results of queries that return schema elements.

In this paper we elaborate on the problem of ranking schema elements. We start by introducing and discussing a number of metrics (mainly originating from the area of Information Retrieval) which could be used for evaluating automatic methods for ranking schema elements. Subsequently, we introduce, and formally

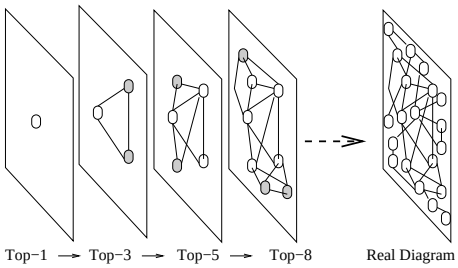

Figure 1: Progressive visualization based on top-k diagrams

evaluate, several possible ranking methods. This is another rather unique characteristic of our work, as most (probably all), other works on related topics, do not follow any formal evaluation method. Subsequently, we discuss how we have exploited these ranking methods for offering visualizations that can help users understand quickly an ontology. Specifically, the derivable top-k lists and top-k diagrams allow exploring an ontology gradually: from the more important elements to the less (Figure 1 illustrates the idea).

Although we confine ourselves to RDF schemas [Brickley and Guha 1999], most of the material and results presented in this paper could be applied on any object oriented schema, i.e. a schema defined using the classical o-o structuring mechanisms: classification (objects and classes), attribution, associations (between classes), and generalization/specialization (among classes and among associations).

This paper is organized as follows. Section 2 proposes and discusses metrics for evaluating schema elements ranking methods and Section 3 describes the test collection that we have built. Section 4 defines RDF schemas and introduces notations, and Section 5 defines several ranking methods for RDF schemas. Section 6 reports the results of evaluating these ranking methods on the test collection. Section 7 describes an application of ranking methods on visualization and discusses related work. Finally Section 8 concludes the paper and identifies issues for further research.

2 On Evaluating Ranking Methods for Schema Elements

Ranking structured data and knowledge is a relatively new task which becomes more and more important. However, most of the works around this topic lack a formal evaluation methodology. In this section we elaborate on various metrics that could be used for evaluating automatic methods for ranking schema elements.

2.1 Evaluation Metrics

One approach to evaluate a ranking method is to suppose that for each schema there is an *ideal ranking* of its classes according to their importance. This ranking could be provided by an "oracle" or an expert (i.e. a person that knows well the schema and can provide us with such a ranking), or by aggregating the rankings provided by several persons. In the latter case, the aggregated ranking can be obtained according to various different methods (mainly coming from the area of Social Choice), like plurality ranking, Borda [de Borda 1781] ranking, Condorcet [de Condorcet 1785] ranking or Kemeny Optimal Aggregation [Kemeny 1959], but we shouldn't forget the Arrow's impossibility theorem [Arrow 1951].

Let $C = \{c_1, ..., c_N\}$. Any total and one-to-one function $f : [1..N] \rightarrow [1..N]$ defines a linear ordering of C. In particular, given two classes c_i and c_j, if $f(i) < f(j)$ then f ranks c_i higher than c_j. We can denote this ranking by C_f and write $C_f = \langle c_{f(1)}, ..., c_{f(N)} \rangle$.

We shall use the notation $C_f(k)$ to denote the first k elements ($k \leq N$) of the ranking C_f, i.e. $C_f(k) = \langle c_{f(1)}, ..., c_{f(k)} \rangle$. We shall also use $C_f\{k\}$ to denote the unordered set of elements that appear in $C_f(k)$, i.e. $C_f\{k\} = \{c_{f(1)}, ..., c_{f(k)}\}$.

Let's denote the ideal ranking by I, i.e. $C_I = \langle c_{I(1)}, ...c_{I(N)} \rangle$. What we need is a method for comparing an automatically derived ranking $C_A = \langle c_{A(1)}, ...c_{A(N)} \rangle$ with respect to C_I. Below we introduce and discuss a number of metrics that can be used for our purposes which are inspired from the measures used in the area of Information Retrieval (IR).

– *k-Precision*

  One metric used in IR, is the R-precision, where R stands for the number of documents in the test (evaluation) collection that is known to be relevant to the evaluation query. It is a single number metric which, in contrast to other measures (like precision, recall, E-measure), somehow takes into account the order of the documents returned by the system (because it considers only the R first elements returned by the system). In our case, in place of R we can use any k that is appropriate to our needs and thus define:

$$sim_p(I, A, k) = \frac{|C_I\{k\} \cap C_A\{k\}|}{k}$$

– 1..*k-Precision*

  Consider two rankings C_A and C_B whose first k elements contain exactly the first k elements of the ideal ranking, i.e. $C_A\{k\} = C_B\{k\} = C_I\{k\}$. Now assume that $C_A(k) = C_I(k)$, that is, they have the same order, while $C_B(k) \neq C_I(k)$ (e.g. $C_B(k)$ could be the reverse order of $C_I(k)$). Although A is definitely better than B, this cannot be identified with the k-precision, because $sim_p(I, A, k) = sim_p(I, B, k) = 1$. To tackle such cases we can

use k measures instead of one, specifically the measures 1-precision, ... , k-precision. So we can define:

$$sim_{pk}(I, A, k) = (sim_p(I, A, 1), ..., sim_p(I, A, k))$$

The results of comparing two rankings according to this k-valued measure can be illustrated by plotting the interpolating curve for each of the two sets of k values. The higher a curve (in the Y-axis) is, the better.

– PR-curves

We could also define a measure analog to Precision/Recall curves of IR. However, here we have to decide which are the "relevant" elements. We can consider as relevant elements the first k elements of I. It is necessary to choose a $k < N$, otherwise the resulting PR-curves will coincide with the function $f(x) = 1$. Suppose we have selected such a k. The curve is defined as follows: we start scanning C_A and when we meet the first element, say at position i, that belongs to $C_I\{k\}$ we compute the precision at that position, i.e. the value $1/i$, when we find the second, say at position i' we compute the value $2/i'$, and so on. We continue in this way until we have consumed the entire C_A. We can then plot the interpolating curve of the values found.

Notice the differences with 1..k-precision: In 1..k-precision the nominator is $|C_I\{i\} \cap C_A\{i\}|$ while here it is $|C_I\{k\} \cap C_A\{i\}|$. One benefit of the latter is that $C_I\{k\}$ is more "reliable" than $C_I(k)$. We made this observation while building our test collection (which is discussed in detail at Section 3). Specifically, we noticed that experts could easily select the major 20 classes of a schema, however, it was not easy for them to order these 20 classes in a principled way (the ordering was done rather arbitrarily). Consequently, the results of an evaluation based on PR-curves is safer than those based on 1..k-precision. Finally, an obvious difference from 1..k-precision is that here we compute the precision only when a new relevant element occurs (and not at every $i = 1..k$).

– Recall/Fallout curves

Instead of PR-curves, we can employ Recall-Fallout graphs. In IR, fallout is defined as the proportion of the non-relevant documents retrieved. In our case, and with the assumption that the first k elements of I are considered as the relevant ones, we can define fallout as follows

$$Fallout = \frac{|(C - C_I\{k\}) \cap C_A\{k\}|}{N - k}$$

The evaluation with RF-graphs has some theoretical advantages (over evaluation with PR-curves). Moreover RF-graphs could also be exploited practi-

cally, i.e. for improving the effectiveness of the under-evaluation system (for more see [Robertson 2007]).

– Kemeny distance

One single measure that takes into account the order of the elements is Kemeny distance [Kemeny 1959]. According to Kemeny, the distance between two rankings equals the number of pair-wise disagreements between them.

$$d_{knm}(A,B) = |\{(i,j) \mid A(i) > A(j) \text{ and } B(i) < B(j)\}|$$

This measure can be applied directly if we want to compare the entire rankings. If we want to compare only $C_A(k)$ and $C_I(k)$, one problem is that they may not have the same elements, i.e. it can be $C_A\{k\} \neq C_I\{k\}$. One method to overcome this problem is to make the assumption that an element that is not present in $C_A\{k\}$ resides on the $k+1$ position of C_A (like in [Tzitzikas 2001]). Formally, if $c_i \notin C_A\{k\}$, we can assume that $A(k+1) = i$. We can make exactly the same assumption if we don't know the entire C_I but only $C_I(k)$. This is very convenient for building a test collection as it is much more easy for an expert to come up with an ordered list of the top 20 elements, than an ordered list of 100 elements. Also note that we can apply this measure also to weakly ordered sets.

As all of the above metrics contain expressions of the form $|C_I\{k\} \cap C_A\{k\}|$, it is important to be able to compute these values with accuracy. This is not always possible because a ranking method may produce a lot of ties (especially if the schema is large). This is because there are not so many things (graph features) that can be used to discriminate classes. It follows that a ranking method for schemas is expected to yield more ties than a ranking method for documents because in the latter case the documents are characterized by several words and consequently several numerical values (e.g. according to the TF-IDF weighting scheme). So it is more safe to assume that a ranking method will derive *weakly ordered sets* (else called bucket orders, or rankings with ties). For instance, assume that $C_A = \langle c_1, \{c_2, c_3\}, c_4\rangle$, meaning that c_2 and c_3 are equally ranked. This means that $C_A(2)$ is either $\langle c_1, c_2\rangle$, or $\langle c_1, c_3\rangle$, and that $C_A\{2\}$ is either $\{c_1, c_2\}$, or $\{c_1, c_3\}$. It follows that we cannot compute expressions of the form $|C_I\{k\} \cap C_A\{k\}|$ with accuracy. Also notice that the ideal ranking could be a weakly-ordered set too.

We can rectify this problem by adopting an approach similar to that of the Expected Search Length (proposed in [Cooper 1968]). Specifically, we can consider that every possible linear order of a weakly-ordered set is equiprobable and then compute the "expected" size of $|C_I\{k\} \cap C_A\{k\}|$. For instance, assume that $C_A = \langle c_1, \{c_2, c_3\}, c_4\rangle$ and $C_B = \langle c_2, \{c_4, c_1\}, c_3\rangle$, and suppose that we want to compute $|C_A\{2\} \cap C_B\{2\}|$. It follows that $C_A\{2\} \cap C_B\{2\}$ can be one of the

Schema	$\|C\|$	$\|P\|$ no attributes	$\|Attrs\|$	$\|C\|/\|P\|$
CIDOC	78	243	7	0.312
PhOntology	181	125	7	1.37
Wine Ontology	76	17	0	4.47

Table 1: Schema Features of the Test Collection

following: $\{c_2\}$, $\emptyset$, $\{c_1, c_2\}$, $\{c_1\}$. So the expected value of $|C_A\{2\} \cap C_B\{2\}|$ is $(1+0+2+1)/4 = 1$. This method can be followed to all the above metrics.

3 Building a Test Collection

Here we describe and discuss how we have built a test collection. We selected three RDF schemas:

(a) CIDOC CRM [1]. It is the international standard (ISO 21127:2006) for controlled exchange of cultural heritage information.

(b) PhOntology [2]. It is a top level ontology.

(c) Wine Ontology [3]. It represents knowledge about wines and it is one of the w3c example ontologies.

We selected these three schemas for several reasons. Firstly, we need middle sized schemas, because for small ones (e.g., with 20 classes) the problem of finding top-k classes looses its interest, while for big ones (e.g., with 500 classes) the production of the ideal ranking by experts would be a very difficult task. Additionally, we would like to experiment on schemas of various values of $\frac{|C|}{|P|}$, where $|C|$, $|P|$ is the number of classes/properties respectively. So we selected a schema (CIDOC CRM) with $|C| << |P|$, i.e., a rich ontology, another (PhOntology) with $|C| \simeq |P|$ and finally one (Wine Ontology) with $|C| >> |P|$, i.e., a schema that can be considered more as a taxonomy rather than an ontology.

In this manner we can observe how different ranking methods are affected by the kind of a schema (i.e., whether it is a rich ontology, a taxonomy or something in between). Although building of a bigger test collection is a challenging task, the three schemas selected here can be exploited for a first study (intended by this paper) on ranking RDF schema elements.

Table 1 shows the number of classes, properties and attributes (properties pointing to literal types) of each schema of the test collection.

[1] `http://cidoc.ics.forth.gr/`
[2] `http://athena.ics.forth.gr:9090/RDF/VRP/Examples/phOntology.rdf`, `http://glotta.ntua.gr/nlp/StateoftheArt/Ontologies/phOntology.html`
[3] `http://www.w3.org/TR/owl-guide/wine.rdf`

Note that in general, an RDF schema may reuse and extend elements from other schemas. Our test collection comprises schemas that do not import any other schema[4].

3.1 Deriving Ideal Rankings

Here we describe how we derived the ideal ranking of each schema. We refer to CIDOC as an indicative example of the procedure followed for all schemas.

Each of the four persons involved was asked to study the schema and then select the 10 (and 20) more important classes of the schema. In particular, we asked them to select the 10 classes which according to their opinion would aid them to understand the schema if they did not know anything about it. As the selection of the more important classes is not a "well-defined" task, the evaluators were not feeling comfortable at the beginning of the ranking process. Each of the evaluators was free to follow his own process on coming up with the top-10 or top-20 classes of the CIDOC ontology. Each person needed approximately 40 minutes to complete this task. The ontology was available in the form of a printout containing a list of classes with the superclass of each and a separate printout containing a table with the domain and range of the properties. It was also available electronically in *Protégé* [Noy et al. 2001] and visually using its Jambalaya viewer.

Two out of four persons begun (each one separately) by excluding the less significant classes in their opinion until they reached the 20 most important for them. Then they continued until reaching the 10 most important. The other two (one of them is actually the chair of the CIDOC CRM committee) decided on the top-10 first and then went on to pick the top-20. All noticed that, although they could easily select the major 20 classes of the schema, it was not so easy for them to order these 20 classes in a principled way.

The union of the top-20 classes of the four experts contained 36 classes. We aggregated the results using the Borda method [de Borda 1781]. We assumed that each input has two blocks $\langle\{top11\}, \{top12-20\}\rangle$. The resulting aggregated weak order (comprising 5 blocks) is shown at the left column of Table 2.

The middle and right column of Table 2 shows the result of an analogous process on the PhOntology and WineOntology. The evaluators of these ontologies were not those that evaluated CIDOC.

4 RDF Schemas

Definition 1. *The graph of an RDF schema is a graph* $\Gamma = (\{C \cup L\}, P, SC, SP)$ *where* C *is a set of nodes labeled with a class name,* L *is a set of nodes labeled*

[4] Although the Wine ontology imports the Food schema, there are only 3 classes that extend classes of the Food Ontology.

	CIDOC
1	E39.Actor
2.1	E4.Period
2.2	E5.Event
2.3	E7.Activity
2.4	E18.Physical_Stuff
2.5	E19.Physical_Object
2.6	E53.Place
3.1	E21.Person
3.2	E28.Conceptual_Object
3.3	E55.Type
3.4	E73.Information_Object
4.1	E26.Physical_Feature
4.2	E41.Appellation
4.3	E70.Stuff
4.4	E71.Man-Made_Stuff
5.1	E22.Man-Made_Object
5.2	E24.Physical_Man-Made_St.
5.3	E52.Time-Span
5.4	E72.Legal_Object
5.5	E77.Persistent_Item

	PhOntology
1	Process
2.1	Role
2.2	Description
3	Collection
4	Situation
5	SpatialEntity
6	CausalEntity
7.1	TimeMeasure
7.2	ConstrainedRelationClass
7.3	KnowledgeEngineering
8	Entity
9	Documentation
10	PhysicalEntity
11	AttributeOrMeasure
12.1	EntityPlayingSomeRole
12.2	Repository
13	DescriptionContent
14	Language
15	Protocol
16	ProceduralLanguage

	Wine
1	Wine
2.1	WineTaste
2.2	WineDescriptor
3	Bordeaux
4	Loire
5	RedWine
6	WhiteWine
7	VintageYear
8	Burgundy
9	Vintage
10	Medoc
11	WineGrape
12	WineColor
13.1	WineDescriptor
13.2	DessertWine
14	DryWine
15	SemillonOrSauvBlanc
16	LateHarvest
17.1	Region
17.2	TableWine

Table 2: The ideal rankings of the test collection

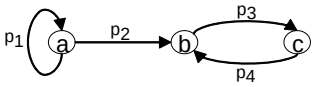

Figure 2: An example of an RDF schema

with a data type (literals), SC is a binary relation over C (subsumption between classes), P is a set of arcs of the form $\langle c_1, p, c_2 \rangle$ *where* $c_1 \in C$, $c_2 \in C \cup L$, *and p is a property name, and finally SP is a binary relation over P (subsumption between properties).*

We consider only schemas that are valid according to the RDF semantics [Hayes 2004]. If $\langle c_1, p, c2 \rangle \in P$ we shall write $domain(p) = c_1$ and $range(p) = c_2$. Table 3 introduces notations that we shall use in the sequel. Roughly, for a given $c \in C$, we use $conn(c)$ to denote those classes which are connected with c through a property. More precisely, we consider only user defined classes, so the elements of $conn(c)$ are always members of C. We use $\uplus$ to denote bag union (so the operands as well as the result of this operation can contain duplicates), e.g. $[a, b, c] \uplus [a] = [a, a, b, c]$. For instance, consider the schema drawn in Figure 2 with three classes a, b, c, and four properties: $\langle a, p_1, a \rangle$, $\langle a, p_2, b \rangle$, $\langle b, p_3, c \rangle$, $\langle c, p_4, b \rangle$. In this case we have: $conn(a) = [a, a, b]$, $conn(b) = [a, c, c]$, $conn(c) = [b, b]$. Given a binary relation R, we use $Tr(R)$ to denote its transitive closure. The bottom part of Table 3 contains notations that take into account the semantics of RDF.

We shall also use $Attrs$ to denote the set of all properties whose range is a literal type (i.e. $Attrs = \cup\{\ attrs(c) \mid c \in C\}$).

Notation	Meaning
$sub(c) = \{c' \mid (c, c') \in SC\}$	
$sup(c) = \{c' \mid (c', c) \in SC\}$	
$subp(c) = sub(c) \cup \text{sup}(c)$	
$sub(p) = \{p' \mid (p, p') \in SP\}$	
$sup(p) = \{p' \mid (p', p) \in SP\}$	
$propFrom(c) = \{p \mid domain(p) = c\}$	
$propTo(c) = \{p \mid range(p) = c\}$	
$prop(c) = propFrom(c) \cup propTo(c)$	
$attrs(c) = \{p \mid from(p) = c \text{ and } to(p) \in L\}$	
$conn(c) = (\uplus\{ range(p) \cap C \mid p \in propFrom(c)\}) \uplus (\uplus\{ domain(p) \mid p \in propTo(c)\})$	
$subAll(c) = \{c' \mid (c, c') \in Tr(SC)\}$	
$supAll(c) = \{c' \mid (c', c) \in Tr(SC)\}$	
$propAllFrom(c) = \{p \mid domain(p) \in \{c\} \cup supAll(c)\}$	
$propAllTo(c) = \{p \mid range(p) \in \{c\} \cup supAll(c)\}$	
$propAll(c) = propAllFrom(c) \cup propAllTo(c)$	
$attrsAll(c) = \{p \mid from(p) \in \{c\} \cup supAll(c) \text{ and } to(p) \in L\}$	

Table 3: RDF-related Notations

5 Ranking Methods for RDF Schemas

Below we introduce a number of methods for ranking RDF schema classes.

$$(m_0): \ R(c) = |sub(c)| + |sup(c)| + |prop(c)|$$

This is the most simplistic method. It views the schema graph as if it was a plain graph (i.e. it does not distinguish arc types).

$$(m_1): \ R(c) = |sub(c)| + |sup(c)| + |propFrom(c)|$$

This is a variation of (m_0) that considers only the properties that define as domain the given class.

$$(m_2): \ R(c) = q + (1-q) \sum_{c' \in conn(c)} \frac{R(c')}{|conn(c')|}$$

If we view classes as states, and properties as bidirectional transitions, then we can realize that according to this method, the score of a class is equal to the stationary probability that a random surfer is at that class[5]. Note that this method ignores properties pointing to literal types, as well as the generalization/specialization hierarchies.

$$(m_3): \ R(c) = q\frac{attrs(c)}{|Attrs|} + (1-q) \sum_{c' \in conn(c)} \frac{R(c')}{|conn(c')|}$$

This is variation of (m_2) that also considers the properties that point to literal types. Specifically, here the probability of reaching a class after a random jump

[5] With probability q the surfer can jump to a randomly selected class while with probability $1-q$ he selects to follow one of the property-derived transitions.

is not the same for all classes. The more attributes a class has, the more probable a surfer jumps to that class. Again, generalization/specialization hierarchies are ignored.

(m_4) :

$$R(c) = q_1 \frac{attrs(c)}{|Attrs|} + q_2 \sum_{c' \in subp(c)} \frac{R(c')}{|subp(c')|} + (1 - q_1 - q_2) \sum_{c' \in conn(c)} \frac{R(c')}{|conn(c')|}$$

This is variation of (m_3) that takes into account the generalization/specialization relationships among classes. Each generalization relationship is viewed as a bidirectional transition. With probability q_1 the surfer jumps to a random class (with preference to those holding more attributes), with probability q_2 he selects to follow one of the isa-derived transitions, while with probability $1 - q_1 - q_2$ he selects to follow one of the property-derived transitions.

$$(m_5): \; R(c) = |subAll(c)| + |supAll(c)| + |propAll(c)|$$

This is actually a variation of (m_1). It is like viewing the "completed" graph, i.e. the graph that particularizes all inherited features (superclasses, subclasses, property domains and ranges), as a plain graph.

It is evident that (m_3) is a special case of (m_4) (corresponding to $q_2 = 0$). Similarly, (m_2) is also a special case of (m_4).

An alternative approach to rank classes, would be to first score properties and then define the scores or the ranking of the classes. One approach to define the significance of properties is to exploit the definition of betweenness centrality of edges [Freeman 1977, Freeman 1979] in a graph. According to this definition, an edge is more important than another, if it appears in more shortest paths between nodes than the other. Formally, let $G : (V, E)$ be a graph. Then, the edge betweenness centrality $BC(e)$ of an edge $e \in E$ is defined as: $BC(e) = \sum_{u,w \in V, u \neq w} \frac{\sigma_{u,w}(e)}{\sigma_{u,w}}$, where $\sigma_{u,w}(e)$ denotes the total number of shortest paths between u and w that pass through edge e and $\sigma_{u,w}$ denotes the total number of shortest paths between u and w.

Note that to compute this metric we can exclude from the schemas multiple-edges because they do not affect the measure of betweenness centrality. It is evident that in this way we will actually rank the distinct pairs of classes rather than the properties. Below we present two methods for ranking classes based on the scores of the properties.

$$(m_6): \; R(c) = \sum_{p \in prop(c)} BC(p)$$

Here the score of a class c is defined as the sum of the BC scores of the properties that have c as domain or range.

An alternative approach is to define top-k classes as the classes that are encountered as domain or range of the top-l properties, where l is the smallest integer. We can call this method (m_7).

6 Experimental Evaluation

We decided to evaluate the ranking methods (m_0), (m_1), (m_4), (m_5), (m_6) and (m_7). As noticed in Section 5, (m_2) and (m_3) are special cases of (m_4) and thus, they are considered in the experimental evaluation through the appropriate values of the parameters of (m_4). For computing (m_4) scores we employed an iterative algorithm with at most 100 iterations[6].

We decided to evaluate the above ranking formulas according to the ideal ranking because this method is less subjective and its results are repeatable. Moreover the ideal ranking can serve as the baseline for comparing different ranking methods. Concerning evaluation metrics, we decided to employ the k-precision metric for various values of k. The selection of the values of k was based on the ideal ranking of the schema at hand. For instance, for CIDOC CRM we used the values 1, 7, 11, 15, 20 because these values follow the block sizes of the ideal ranking (see Table 2). As the points are not so many, we will present the results in a tabular form and not in the form of curves. However, as an automatic method for ranking may yield ties (equally scored classes), the computation of the k-precision was based on Expected Search Length (recall Section 2) so that to obtain more precise results.

We should notice that (m_5) gave by far the worst results. This is due to the fact that (m_5) always favors the classes located deeply in the class hierarchies. These classes inherit a big number of properties from their ancestors. However, psycholinguistic evidence has shown that middle level concepts tend to be more detailed and prototypical of their categories than classes at lower hierarchical levels [Rosch 1978]. This fact has also been experimentally verified for RDF schemas in [Theoharis 2007]. Moreover, methods (m_6), (m_7) gave always worse (or rarely the same) results than (m_0), (m_1), (m_4). In the sequel, for each of the three schemas of our collection we only report the results of (m_5), (m_6) and (m_7) without further discussion. Instead, we focus on the comparison of the rest methods (i.e., m_0, m_1, m_4).

6.1 Schema: CIDOC CRM

Table 4 shows the results of evaluating our ranking methods using 11 and 20-precision.

[6] In general, the number of iterations required for convergence is empirically $\mathcal{O}(\log n)$ where n is the number of links.

Method (q_1, q_2, q_3)	11-precision	20-precision
(m_4) $(q_1, 0.0, 1-q_1)$	0.63 - 0.72	0.6 - 0.7
(m_4) $(q_1, 0.1, 0.9-q_1)$	0.54	0.65 - 0.75
(m_4) $(q_1, 0.2, 0.8-q_1)$	0.45 - 0.54	0.65 - 0.75
(m_4) $(q_1, 0.3, 0.7-q_1)$	0.45 - 0.54	0.65 - 0.7
(m_4) $(q_1, 0.4, 0.6-q_1)$	0.36 - 0.54	0.65 - 0.7
(m_4) $(q_1, 0.5, 0.5-q_1)$	0.45 - 0.54	0.65 - 0.7
(m_4) $(q_1, 0.6, 0.4-q_1)$	0.45	0.7
(m_4) $(q_1, 0.7, 0.3-q_1)$	0.45	0.7
(m_0)	0.63	0.71
(m_1)	0.63	0.7
(m_5)	0.09	0.2
(m_6)	0.63	0.61
(m_7)	0.57	0.6

Table 4: Results of the Evaluation on CIDOC CRM

Let us now analyze in more detail (m_4). Recall that:

(m_4) :

$$R(c) = q_1 \frac{attrs(c)}{|Attrs|} + q_2 \sum_{c' \in subp(c)} \frac{R(c')}{|subp(c')|} + (1 - q_1 - q_2) \sum_{c' \in conn(c)} \frac{R(c')}{|conn(c')|}$$

Note that q_3 in Table 4 corresponds to the value $1-q_1-q_2$. For instance, the first row of Table 4 corresponds to m_4 where q_2 is considered as constant (more precisely $q_2 = 0.0$) and q_1 as variable that takes values in $\{n \times 0.1 \mid n \in Z, 0 \leq n \leq 10\}$. Therefore, $q_3 = 1 - q_1 - q_2 = 1 - q_1$. The derived 11-Precision (resp. 20-Precision) lies in $[0.63, 0.72]$ (resp. $[0.6, 0.7]$) depending on q_1. As the above ranking method is parametric, below we present some figures that illustrate the precision results for various combinations of q_1, q_2, q_3 values. The x-axis of each figure has various values of q_1 $(0, 0.1, 0.2, \ldots 1)$, while the y-axis corresponds to precision values [0,1]. Figure 3 shows the corresponding plots for 7 and 11 precision, while Figure 4 shows the corresponding plots for 15 and 20 precision. Each figure contains a plot for various values of q_2 (the values of q_3 are not shown, as they are derived from the rest values, i.e. $q_3 = 1 - q_1 - q_2$).

From these we observe that the best k-precision values are obtained for low values of q_2. Moreover, we observe that they are obtained for $q_3 > q_1$, i.e., associations are more significant than attributes. Specifically, the combinations (q_1, q_2, q_3) that give the highest 11-precision are $([0.1-0.4], 0.0, 1-q1)$. The corresponding combinations for 20-precision are $(0.1, 0.2, 0.7)$, $([0.4-0.5], 0.1, 0.9-q_1)$.

It is interesting to note that method (m_0) is almost as good as (m_4): although it does not give the maximum values (i.e. 0.72, 0.75 for top-11 and 20 resp.), it gives very close values (i.e. 0.63, 0.71 for top-11 and 20 resp.). In addition, there are several combinations of q_1, q_2 values that make (m_4) to give much worse than (m_0) results. The recall/fallout curve of methods m_0 and m_4 (Figure 5) can provide a means to compare these two methods. As we can see from the curve, (m_0) is slightly better than (m_4), since (m_0) usually lies closer to the

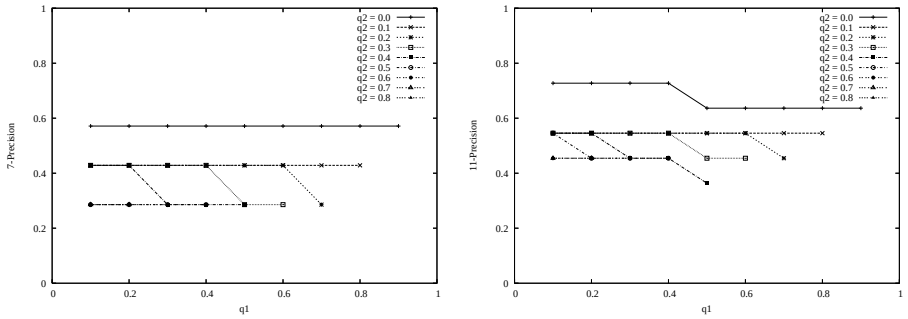

Figure 3: Results for 7 (left) and 11 (right) precision on CIDOC CRM

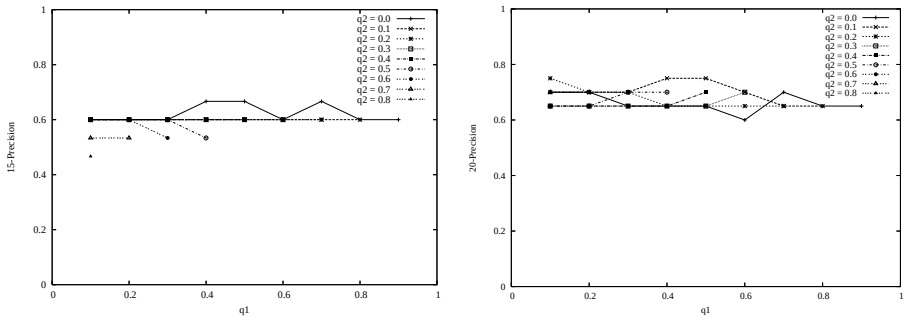

Figure 4: Results for 15 (left) and 20 (right) precision on CIDOC CRM

upper-left corner of the curve (where the recall is 1.0 and the fallout is 0.0) than (m_4).

We should note that each association of the CIDOC CRM ontology has been represented in RDFS as two counterpoising properties between the involved pair of classes (each labeled with a different name). This fact does not hold for attributes, i.e., properties pointing to Literal class, since Literal cannot be the domain of any property. This means that in this ontology, ranking classes according to $prop(c)$ is almost equivalent to ranking them according to $propFrom(c)$ or $propTo(c)$.

Finally, Table 5 shows the top-20 distinct pairs of classes of CIDOC CRM.

6.2 Schema: PhOntology

Table 6 shows the results of evaluating our ranking methods using 10 and 20-precision.

Here, method (m_0) gives the best results ((m_0) yields much higher 10-precision and slightly lower 20-precision than (m_1)). Concerning (m_4), the

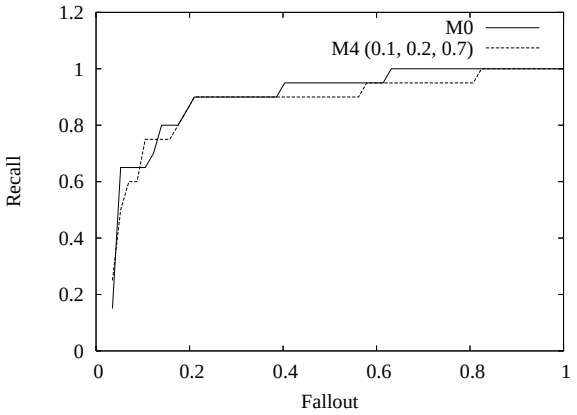

Figure 5: The Recall/Fallout curve of (m_0) and (m_4) on CIDOC CRM

k	Pair of Classes
1	(E39.Actor, E7.Activity)
2	(E5.Event, E39.Actor)
3	(E77.Persistent_Item, E5.Event)
4	(E39.Actor, E18.Physical_Stuff)
5	(E1.CRM_Entity, Literal)
6	(E19.Physical_Object, Literal)
7	(E7.Activity, E1.CRM_Entity)
8	(E19.Physical_Object, E53.Place)
9	(E53.Place, E39.Actor)
10	(E74.Group, E39.Actor)
11	(E52.Time-Span, Literal)
12	(E18.Physical_Stuff, E53.Place)
13	(E70.Stuff, E7.Activity)
14	(E1.CRM_Entity, E24.Physical_Man-Made_Stuff)
15	(E54.Dimension, E70.Stuff)
16	(E57.Material, E18.Physical_Stuff)
17	(E55.Type, E7.Activity)
18	(E71.Man-Made_Stuff, E7.Activity)
19	(E11.Modification_Event, E55.Type)
20	(E79.Part_Addition, E18.Physical_Stuff)

Table 5: Top-20 pairs of classes of CIDOC CRM

combination of (q_1, q_2, q_3) values that yield the best 10-precision are of the form $(q_1, 0.0, 1 - q_1)$, while the combinations that yield the best 20-precision is (0.5, 0.4, 0.1). More details are given in Figure 6 for the 5- and the 10-precision, and in Figure 7 for the 14- and the 20-precision. We can conclude, that for small values of k the k-precision increases as long as q_2 decreases. However, for high values of k (e.g. k=20), the best k-precision is yielded for high q_2 values, i.e. $q_2 = 0.4$.

Finally, Table 7 shows the top-20 distinct pairs of classes of PhOntology.

Method (q_1, q_2, q_3)	10-precision	20-precision
(m_4) $(q_1, 0.0, 1 - q_1)$	0.6	0.45
(m_4) $(q_1, 0.1, 0.9 - q_1)$	0.5	0.45 - 0.5
(m_4) $(q_1, 0.2, 0.8 - q_1)$	0.4 - 0.5	0.45 - 0.5
(m_4) $(q_1, 0.3, 0.7 - q_1)$	0.4 - 0.5	0.4 - 0.45
(m_4) $(q_1, 0.4, 0.6 - q_1)$	0.4 - 0.5	0.4 - 0.6
(m_4) $(q_1, 0.5, 0.5 - q_1)$	0.3 - 0.5	0.4 - 0.55
(m_4) $(q_1, 0.6, 0.4 - q_1)$	0.3 - 0.4	0.4 - 0.55
(m_4) $(q_1, 0.7, 0.3 - q_1)$	0.3 - 0.4	0.4 - 0.45
(m_0)	0.9	0.79
(m_1)	0.8	0.81
(m_5)	0.18	0.3
(m_6)	0.53	0.4
(m_7)	0.5	0.3

Table 6: Results of the Evaluation on PhOntology

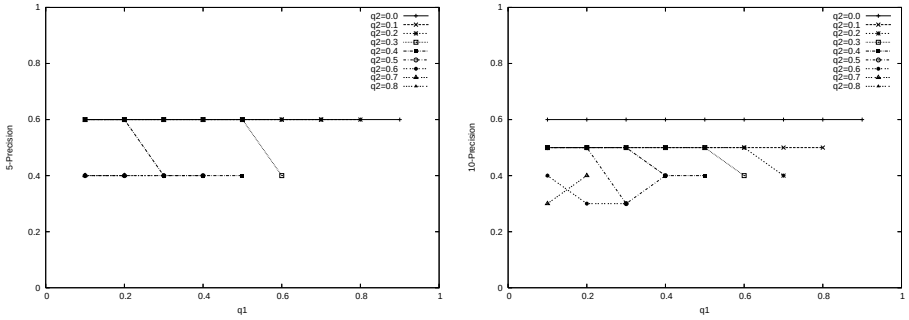

Figure 6: Results for 5 (left) and 10 (right) precision on PhOntology

6.3 Schema: Wine Ontology

Table 8 shows the results of evaluating our ranking methods using 10 and 20-precision. As one can see, (m_1) is better than (m_0). This is due to the fact that Wine Ontology has few properties and the consideration of $|prop(c)|$ (instead of $|propFrom(c)|$) yields many ties. We observe that (m_1) is usually better for 10-precision. For 20-precision (m_4) is better than (m_1) for specific combinations of q_1, q_2, q_3. Specifically, the (q_1, q_2, q_3) combinations that yield the best 10-precision are of the form $([0.1, 0.7], 0.1, 0.9 - q_1)$ and $([0.1, 0.2], 0.2, 0.8 - q_1)$, while the combinations that yield the best 20-precision are of the form $([0.1, 0.7], 0.1, 0.9 - q_1)$ and $([0.1, 0.3], 0.2, 0.8 - q_1)$. Figure 8 shows the corresponding plots for 5 and 10 precision, while Figure 9 shows the corresponding plots for 15 and 20 precision.

Finally, Table 9 shows the top-20 distinct pairs of classes of Wine Ontology.

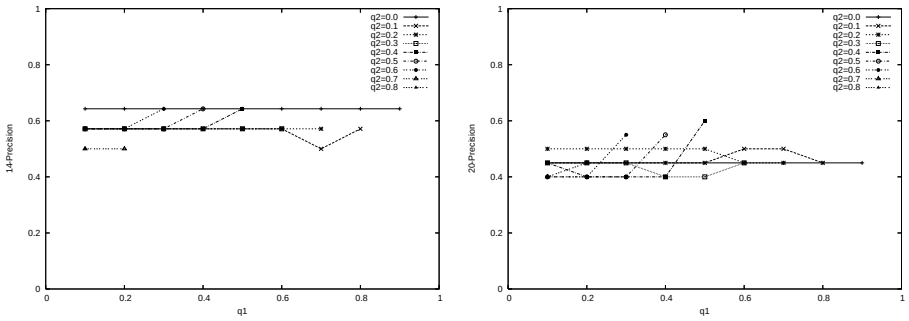

Figure 7: Results for 14 (left) and 20 (right) precision on PhOntology

k	Pair of Classes
1	(Process, SpatialEntity)
2	(Process, Description)
3	(Process, Situation)
4	(Process, Entity)
5	(SpatialEntity, SpatialAttributeOrMeasure)
6	(SpatialEntity, LengthMeasure)
7	(Process, SpeedMeasure)
8	(Description, DescriptionContainer)
9	(Description, ModalityMeasure)
10	(Description, DescriptionMedium)
11	(Description, Boolean)
12	(Process, State)
13	(SpatialEntity, AreaMeasure)
14	(Process, ProcessAttributeOrMeasure)
15	(SpatialEntity, VolumeMeasure)
16	(Process, Event)
17	(Description, TimeMeasure)
18	(Process, CausalEntity)
19	(Situation, TimeMeasure)
20	(Description, CausalEntity)

Table 7: Top-20 pairs of classes of PhOntology

6.4 Summarizing the Results

In most cases, (m_0) gave the best results. (m_4) gave slightly better results only for specific combinations of (q_1, q_2, q_3) for the CIDOC and the Wine Ontology (rows 1, 2 and 3 of Table 4 and rows 3 and 4 of Table 8).

Table 10 shows the average precisions (over our test collection) that were obtained from each method for $k = 7, 11, 20$. For the case of (m_4), the table shows only the combinations of q_1, q_2, q_3 that gave the best results.

Concerning the (m_4) method and comparing with CIDOC and PhOntology schemas, we observe, that in case of Wine Ontology the ignorance of the subsumption properties (i.e. $q_2 = 0.0$) does not yield the best (not even a good) k-precision. This is due to the fact that Wine Ontology consists of few properties and thus the subsumption relationships play a more significant role to the selection of the top-k classes.

Method (q_1, q_2, q_3)	10-precision	20-precision
(m_4) $(q_1, 0.0, 1-q_1)$	0.4	0.4
(m_4) $(q_1, 0.1, 0.9-q_1)$	0.5 - 0.7	0.5 - 0.65
(m_4) $(q_1, 0.2, 0.8-q_1)$	0.5 - 0.7	0.5 - 0.65
(m_4) $(q_1, 0.3, 0.7-q_1)$	0.5 - 0.6	0.5 - 0.6
(m_4) $(q_1, 0.4, 0.6-q_1)$	0.5 - 0.6	0.5 - 0.6
(m_4) $(q_1, 0.5, 0.5-q_1)$	0.5 - 0.6	0.5 - 0.6
(m_4) $(q_1, 0.6, 0.4-q_1)$	0.5 - 0.6	0.5 - 0.6
(m_4) $(q_1, 0.7, 0.3-q_1)$	0.5 - 0.6	0.55 - 0.6
(m_0)	0.21	0.35
(m_1)	0.7	0.53
(m_5)	0.4	0.35
(m_6)	0.25	0.2
(m_7)	0.46	0.35

Table 8: Results of the Evaluation on Wine Ontology

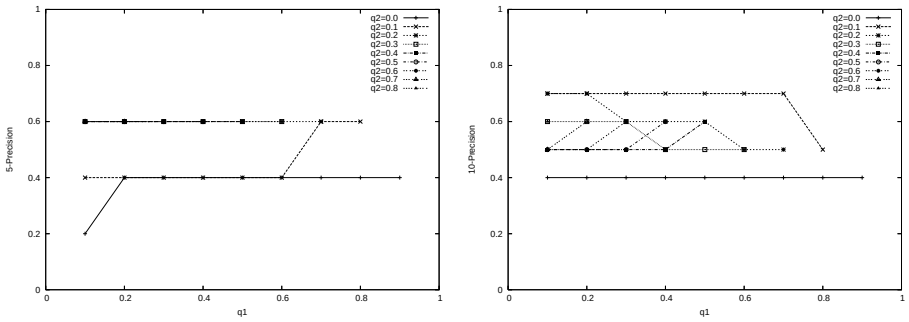

Figure 8: Results for 5 (left) and 10 (right) precision on Wine Ontology

Table 11 presents in detail the rankings of CIDOC CRM yielded by (m_1) and (m_4), with respect to the ideal ranking and the graph features of these classes. Specifically, the first 20 rows (i.e., $1-5.5$) correspond to the top-20 classes of the ideal ranking (see also the left part of Table 2), while the six following rows (i.e., $6.1-8.26$) correspond to classes ranked in the top-20 by (m_1) method (the first column corresponds to the rank of these classes in the ideal ranking, e.g., $E1.CRM\_Entity$ lies in the $6-th$ block of the ideal ranking). For (m_4) we considered the combination $(q_1, q_2, q_3) = (0.1, 0.2, 0.7)$, i.e., one of those that yield the best 20-Precision. For each ranking method we write in bold the ranks of classes that, although included in the top-20 of the ideal ranking, they do not appear in the top-20 of the specific method.

One interesting observation is that there are classes in the top-20 of the ideal ranking that both methods failed to rank in their top-20 (e.g., E21.Person, E26.Physical_Feature, E71.Man-Made_Stuff, E22.Man-Made_Object, E72.Legal _Object). Their common characteristic is the low plain degree value (column plainD in Table 11), that shows that these classes have not been (conceptually) analyzed in detail. A correlated observation is that there are three classes

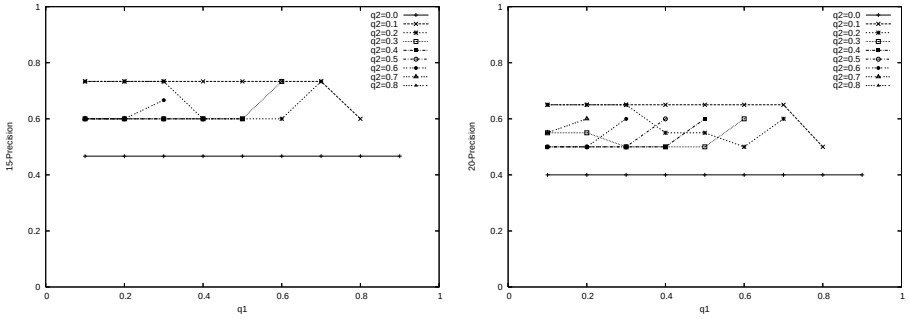

Figure 9: Results for 15 (left) and 20 (right) precision on Wine Ontology

k	Pair of Classes
1	(Array, VintageYear)
2	(Array, WineGrape)
3	(Array, WineColor)
4	(Wine, WineColor)
5	(Array, WineSugar)
6	(Array, WineFlavor)
7	(Array, WineBody)
8	(CotesDOr, WineFlavor)
9	(RedBordeaux, WineGrape)
10	(Array, Winery)
11	(Meursault, WineBody)
12	(RedBurgundy, WineGrape)
13	(PinotNoir, WineColor)
14	(WhiteBurgundy, positiveInteger)
15	(VintageYear, WineGrape)
16	(Meritage, WineColor)
17	(Vintage, VintageYear)
18	(DessertWine, WineSugar)
19	(WhiteBordeaux, WineGrape)
20	(Wine, WineDescriptor)

Table 9: Top-20 pairs of classes of Wine Ontology

(E1.CRM_Entity, E2.Temporal_Entity, E54.Dimension) ranked in the top-20 by both methods that do not appear in the top-20 of the ideal ranking. Their common characteristic is the big plain degree, that shows that they are analyzed in detail. These two observations made us to realize, that the graph features of classes are able to capture the intuition behind the selection of significant classes by experts (who are conceptual modelers) only up to some degree (that fortunately yields satisfactory k-Precisions for our application).

Concerning (m_0) and random walks, a related remark is that in an undirected (strongly connected and non-bipartite) graph $G = (V, R)$, the stationary probability of a node u is given by $P(u) = \frac{deg(u)}{2|R|}$ where $deg(u)$ is the degree of u [Motwani and Raghavan 1995]. One can easily see, that (m_0) corresponds to this case.

	Average precision			
k	(m_0)	(m_1)	(m_4) for (q_1, q_2, q_3)	
7	0.57	0.57	0.57	$(0.3-0.6, 0.2, 0.8-q_1)$ $(0.3-0.6, 0.2, 0.8-q_1)$ $(0.1, 0.4, 0.5)$
11	0.63	0.63	0.60	$(0.1-0.3, 0.1, 0.9-q_1)$
20	0.71	0.7	0.63	$(0.1, 0.2, 0.7)$, $(0.5, 0.4, 0.1)$

Table 10: Average Precision

	CIDOC Class	**propsFrom**	**propsTo**	**subs**	**sups**	**plainD**	m_0	m_1	m_4
1	E39.Actor	17	17	2	1	37	1	1.2	2
2.1	E4.Period	8	8	1	1	18	9	8.2	19
2.2	E5.Event	2	2	3	1	8	**17.1**	12.1	20
2.3	E7.Activity	11	11	7	1	30	5	2	5
2.4	E18.Physical_Stuff	16	16	3	1	36	2	1.3	6
2.5	E19.Physical_Object	10	9	2	1	22	7	5	15
2.6	E53.Place	15	15	0	1	31	4.1	4.2	8
3.1	E21.Person	4	4	0	2	10	15	**12.4**	**27**
3.2	E28.Conceptual_Object	1	1	3	1	6	**19.14**	**13.4**	12
3.3	E55.Type	12	12	3	1	28	6	4.1	4
3.4	E73.Information_Object	5	5	4	2	16	11	7.1	9
4.1	E26.Physical_Feature	1	1	2	1	5	**20.5**	**14.10**	**51**
4.2	E41.Appellation	2	1	6	1	10	14	9.1	18
4.3	E70.Stuff	6	6	2	1	15	12.1	9.2	11
4.4	E71.Man-Made_Stuff	3	3	2	1	9	16.3	**12.3**	**25**
5.1	E22.Man-Made_Object	0	0	1	2	3	**22.4**	**15.14**	**66**
5.2	E24.Physical_Man-Made_Stuff	7	7	3	2	19	8	6	10
5.3	E52.Time-Span	10	6	0	1	17	10	7.2	14
5.4	E72.Legal_Object	2	2	2	1	7	**18.12**	**13.2**	**29**
5.5	E77.Persistent_Item	5	5	4	1	15	12.2	8.1	3
6.1	E1.CRM_Entity	15	14	5	1	35	3	1.1	1
6.2	E2.Temporal_Entity	14	14	2	1	31	4.2	3	7
7.2	E11.Modification_Event	4	4	3	1	12	13.2	10	**24**
7.4	E13.Attribute_Assignment	2	2	4	1	9	16.4	11.2	**22**
7.9	E63.Beginning_of_Existence	1	1	5	1	8	**17.2**	11.3	**23**
8.26	E54.Dimension	6	5	0	1	12	13.1	11.1	17

Table 11: Features of classes of the ideal ranking for CIDOC CRM

6.5 On Generalizing the Results

As our test collection is small, we conducted some additional experiments in order to see whether the results of our comparative evaluation would be different in case we had a bigger test collection. We selected 20 ontologies (those listed at Table 12) and for each one of them we applied the formulas (m_1), (m_0) and (m_4). Then we compared the returned rankings, specifically to compare two ranking formulas A and B, we used the metric $sim(A,B) = \frac{|C_A\{20\} \cap C_B\{20\}|}{20}$. Table 12 shows the results and its last row shows the average values. We can see that the pair $((m_1),(m_0))$ behaves very similarly (similarity 0.9), while the pairs $((m_1),(m_4))$ and $((m_0),(m_4))$ have degree of similarity around 0.7. By making the hypotheses that the 20-precision of (m_0) is around 0.7 (as measured in our test collection), and that the similarity of $((m_0),(m_4))$ is not worse than 0.7 (as measured in this experiment), we could make the conjecture that in general the

	ontology	$sim((m_1),(m_0))$	$sim((m_1),(m_4))$	$sim((m_0),(m_4))$
1	ACM_CSS	1.0	0.33	0.33
2	bsr	0.5	0.8	0.76
3	cerif	0.95	0.45	0.45
4	Datatypes	1.0	0.55	0.55
5	DCD	1.0	0.35	0.35
6	earthrealm	1.0	0.95	0.95
7	GOLD	1.0	0.87	0.87
8	Math_Intern	1.0	0.48	0.48
9	mgentology	1.0	0.95	0.95
10	Mid-Level_Ontology	0.88	0.72	0.75
11	moviedatabase	0.85	0.96	0.95
12	mygrid-services-lite	1.0	0.91	0.91
13	ntua3	1.0	0.78	0.83
14	p3p	0.9	0.58	0.59
15	Phenomenon	0.97	0.93	0.93
16	Property	1.0	0.9	0.9
17	spase_051214	0.5	0.59	0.59
18	substance	1.0	0.57	0.91
19	SUMO	0.8	0.55	0.69
20	tap	1.0	0.57	0.57
AVG		**0.91**	**0.69**	**0.71**

Table 12: Comparing $(m_1), (m_0), (m_4)$

20-precision of (m_0) is expected to be 0.7 ± 0.2.

7 Applications and Related Work

7.1 Visualization

To understand an ontology (from its RDF representation in XML) or to select (from an ontology repository) the ontology that best fits the requirements of an application, is a hard and time consuming task for a human. In this section we show how ranking can be exploited in visualization so that to alleviate this task.

For this purpose we developed a graphical editor for visualizing RDF schemas, called **StarLion** (this tool is part of the RDFSuite [FORTH-ICS 2005]). The graphical layout in the 2D space, is derived by a force-directed placement algorithm (specifically by adapting for the case of RDF schemas the algorithm described in [Tzitzikas and Hainaut 2005]). Figure 10 shows the layout produced for the CIDOC CRM schema. It is evident that such drawings cannot help a user to get acquainted with a schema. From our experiments, we again verified the observation (of [Tzitzikas and Hainaut 2005] for the case of ER diagrams, and of [Theoharis 2007] for the case of RDF schemas), that conceptual schemas tend to have a very connected kernel[7]. This means that it is rather impossible to have a readable (and aesthetically pleasing) 2D layout for medium and large sized schemas; instead we should expect an overwhelming number of edge crossings.

[7] According to [Theoharis 2007], the total-degree distribution of the property graph of Semantic Web schemas usually follows a power law.

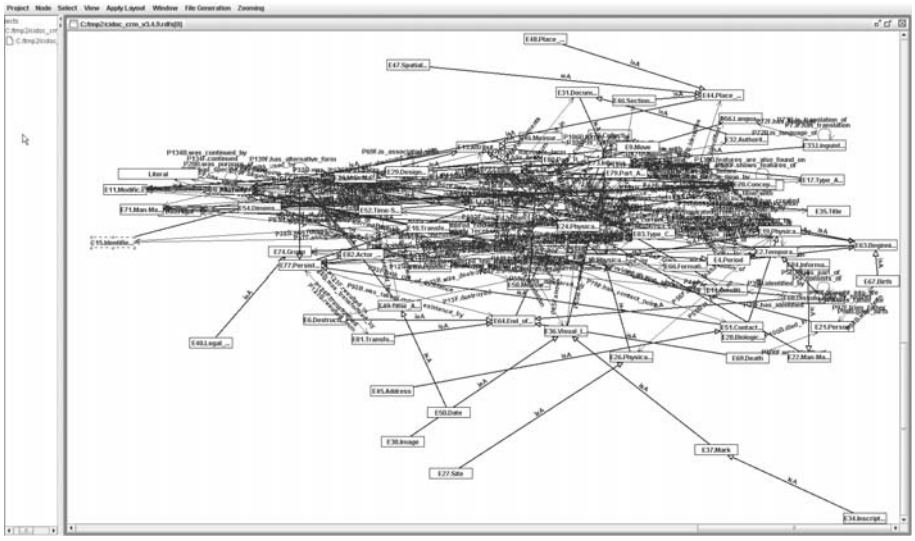

Figure 10: The layout of the entire CIDOC CRM ontology

Ranking could alleviate this problem, as it enables deriving smaller in size (and more readable) diagrams comprising the most important concepts and relationships. The derivable top-k lists and top-k diagrams allow exploring a schema gradually: from the more important elements to the less (Figure 1 illustrates the idea). In addition, and given the inability to produce automatically aesthetically satisfying layouts for large schemas, the small (top-k) diagrams can be very useful for communication purposes. We implemented the ranking methods described above and provided the ability to visualize top-k diagrams for variable values of k. Figure 11 shows the top-7 diagram, Figure 12 shows the top-11, and Figure 13 the top-20 diagram (due to ties this diagram comprises 23 classes) of CIDOC CRM.

Apart from being human readable, these diagrams can be drawn much faster. Note that the force-directed placement algorithm has quadratic time complexity. For instance, the layout for Figure 10 required 25.637 seconds to compute, while the top-7 diagram only 3.024 second. An internal graph representation with double adjacency lists was employed allowing the efficient layout of the top-k diagrams for small values of k, without having to reconstruct the graph. In this way we obtain efficient drawing for values of k up to 20 (which is the most common values).

In addition, StarLion supports a semi-automatic layout process, which can greatly improve the resulting layout. Specifically, the user is allowed to nail down some nodes at desired positions and apply the layout algorithm on the rest part

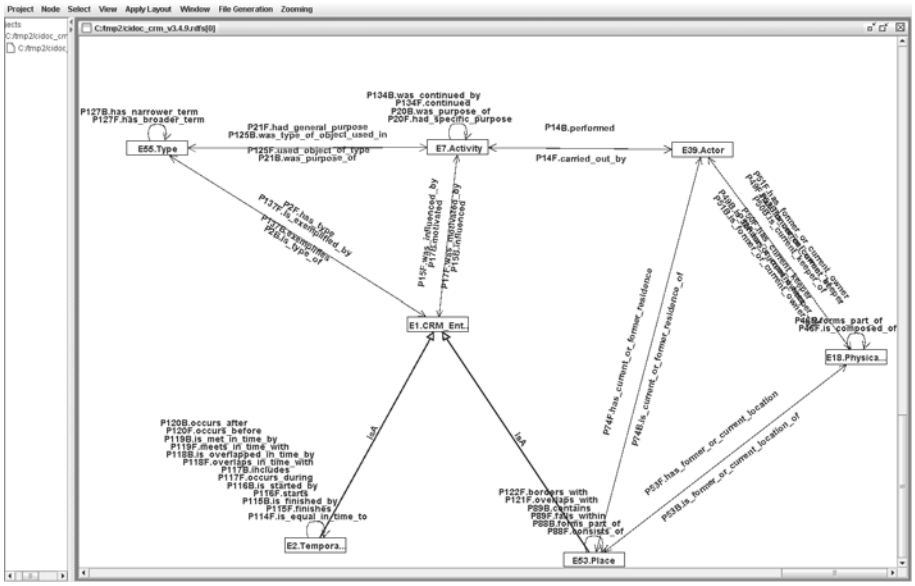

Figure 11: The layout of the top-7 diagram of CIDOC CRM

of the diagram, or the make some elements invisible. The activity diagram of the supported process is sketched in Figure 14. The activity "Select Parameters" allows the user to set the parameters (e.g. spring length and stiffness, magnitude of electric and magnetic repulsion) of the force-directed placement algorithm.

7.1.1 Ranking Properties

For deriving the top-k diagram we first select the top-k classes, and then add all properties among them. Usually, there will be several properties between a given pair of classes. In that case, they are visualized by a single edge and a label for each property. However, one might want to reduce the property labels displayed (e.g. in Figure 11 there is one edge with more than 10 labels). We could try to rank the properties so that to have a criterion to filter them out. One can easily see that if we exploit the scores of classes for defining the scores of properties, then all properties between a given pair of classes will receive the same score, so this approach will not allow us to rank them. For this reason, we propose exploiting the specialization relationships between properties in order to differentiate between them. Specifically, we can define the score of a property p as $score(p) = |subAll(p)|$ where $subAll(p)$ denotes the set of all (direct and indirect) subproperties of p. The higher the score of a property is the higher this property is ranked.

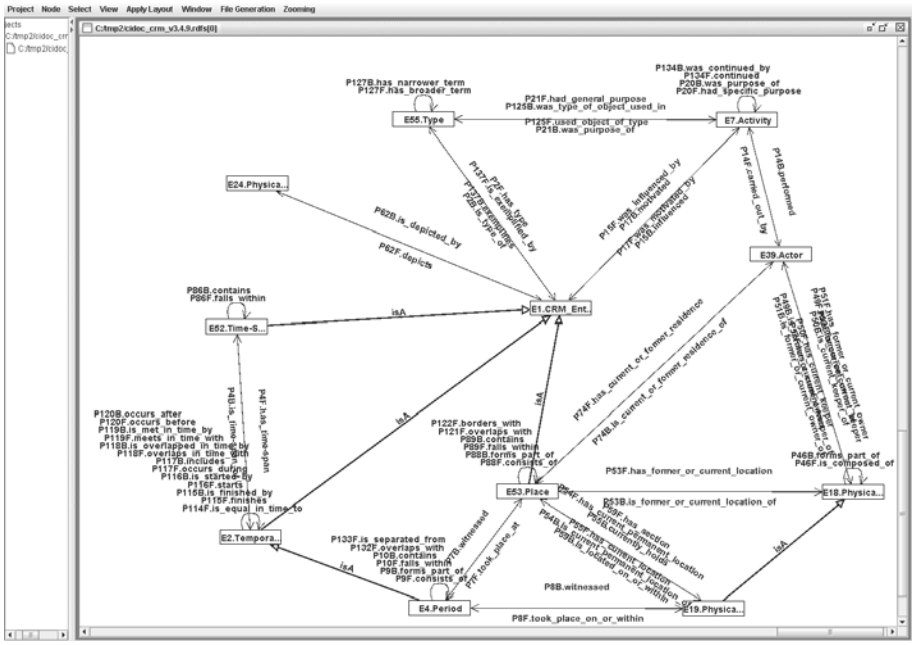

Figure 12: The layout of the top-11 diagram of CIDOC CRM

7.2 Related Work

Most (or probably all) of the work that has been done so far, concerns solely the problem of ontology selection and evaluation, i.e. the problem of ranking a set of ontologies according to various criteria. This task is very useful for Semantic Web search engines and ontology libraries (like Swoogle [Ding et al. 2005], OntoSelect [Buitelaar et al. 2004], OntoKhoj [Patel et al. 2003]). For instance, OntoKhoj [Patel et al. 2003] is an ontology portal that crawls, classifies and searches ontologies. It uses the OntoRank algorithm, a PageRank-like algorithm, which instead of relying on HTML links it relies on the instantiation and subsumption links between ontologies. Analogous approaches (for ranking ontologies, not their constituents) include ActiveRank [Alani and Brewster 2005] and OntoSelect [Buitelaar et al. 2004] (the latter also considers ontology import statements). Swoogle [Ding et al. 2005] is another SW search engine that provides ontology ranking in a way similar to that of OntoKhoj. Although that work also proposes methods for ranking graphs, terms and triples, all these methods are based on the entire network of semantic web ontologies and data. This means that the results obtained by these methods are totally different from ours, as we confine ourselves to the information available in each ontology in isolation (so we do not take into account possible instantiations of the ontology or other interconnected

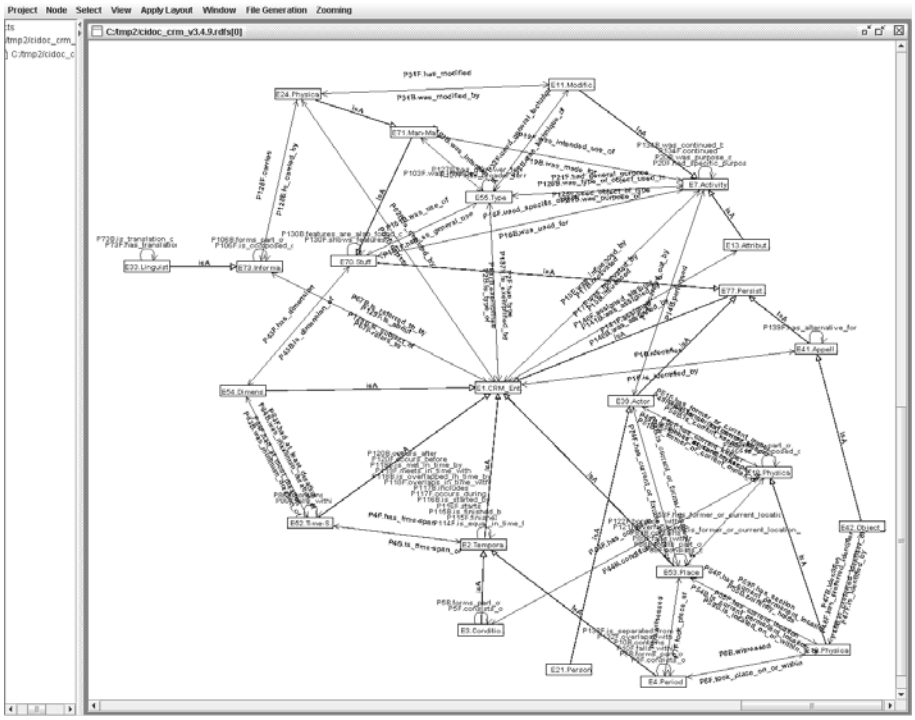

Figure 13: The layout of the top-20 diagram of CIDOC CRM

ontologies). Finally, [Sabou et al. 2006] surveys the works on ontology ranking and tries to relate them with the task of ontology evaluation. Of course, our work could also be exploited by ontology search engines. For instance, each ontology in the result of an ontology seeking query, could be accompanied by its top-k diagram.

Other somehow related works include [Zhuge and Zheng 2003], where PageRank -style ranking formulas for semantic networks are proposed but without any form of evaluation, and [Sheth et al. 2005, Anyanwu et al. 2005] which describes methods for ranking associations and paths among resources (not among schema elements).

Finally we would like to stress the fact that we have conducted an experimental evaluation based on a test collection (to the best of our knowledge no other related work has been evaluated in this way), and this allowed us to realize, the previously unexpected for us fact, that simplistic ranking methods are not worse than complicated ones.

Regarding visualization, we confined ourselves to the case of classical 2D graph plain graph layout algorithms, as our focus in on ranking, not on visual-

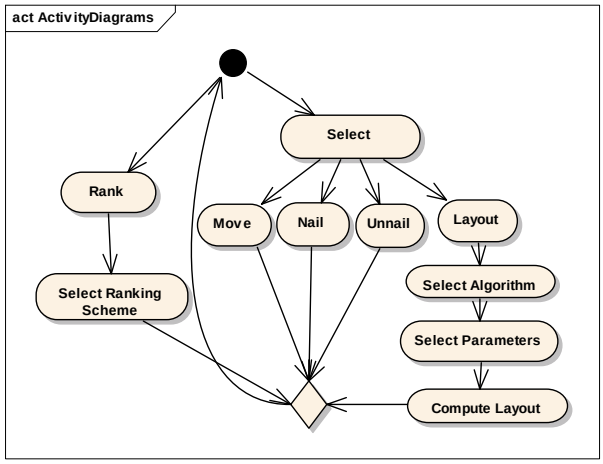

Figure 14: Activity diagram of the semi-automatic layout process supported by StarLion

ization per se. So works like [van Ham et al. 2004] are out of the scope of this paper.

8 Conclusions

The contributions of this paper are: (a) we discuss in detail the issues regarding the evaluation of ranking schema elements and propose specific metrics, (b) we detail the construction of a test collection for ranking RDF schema elements, (c) we introduce and evaluate a number of ranking functions for RDF schema elements and analyze the results, and (d) we report our experiences from exploiting the ranking functions on the problem of ontology visualization.

Concerning the results of the evaluation, although the most closely related works on semantic web ranking, employ iterative PageRank style ranking schemes, our experiments showed that the more simple ranking method (the plain graph degree) gives the best results and it is more stable. Furthermore, it is the fastest to compute method, as it does not require the employment of an iterative algorithm. The average precision obtained for (top-5 to top-10) is about 0.7, which is satisfactory for visualization application purposes.

There are several issues that are worth further research. One of them is the extension of the test collection to a bigger and more representative corpus. Another totally unexplored direction is the definition of appropriate evaluation metrics that take into account both classes and properties. Furthermore, we would like to achieve a deeper understanding of the factors that make conceptual

modelers (experts) to decide that a class is significant. The investigation of which is the maximum precision that could be achieved by a ranking function based only on schema graph features could be useful in order to evaluate ranking methods (e.g., by realizing that a method is optimal). Finally, another direction of research is to devise ranking methods for OWL [Dean et al. 2002] schemas. The question here, is how to best exploit the logical expressions (e.g. unionOf and intersectionOf) that may be used in the definition of a class for ranking purposes.

Acknowledgements

We would like to thank Stergios Leonidis for implementing StarLion and all ranking methods. Many thanks also to Martin Doerr and Dimitris Velegrakis for their participation in deriving the ideal orderings of various schemas.

References

[Alani and Brewster 2005] Alani, H., Brewster, C.: "Ontology ranking based on the analysis of concept structures"; In Procs of the 3rd intern. conf. on Knowledge Capture, K-CAP'05, pages 51–58, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-163-5. http://doi.acm.org/10.1145/1088622.1088633.

[Anyanwu et al. 2005] Anyanwu, K., Maduko, A., and Sheth, A.: "SemRank: ranking complex relationship search results on the semantic web"; Proceedings of the 14th international conference on World Wide Web, pages 117–127, 2005.

[Arrow 1951] Arrow, K. J.: "Social Choice and Individual Values"; 1951.

[Baeza-Yates and Ribeiro-Neto 1999] Baeza-Yates, R., and Ribeiro-Neto, B.: "Modern Information Retrieval"; ACM Press, Addison-Wesley, 1999.

[Brickley and Guha 1999] Brickley, D., and Guha, R. V.: "Resource Description Framework (RDF) Schema specification: Proposed Recommendation, W3C"; March 1999. http://www.w3.org/TR/1999/PR-rdf-schema-19990303.

[Buitelaar et al. 2004] Buitelaar, P., Eigner, T., and Declerck, T.: "OntoSelect: A Dynamic Ontology Library with Support for Ontology Selection"; In Procs of the Demo Session at the International Semantic Web Conference 2004.

[Cooper 1968] Cooper, W.: "Expected Search Length: A Single Measure of Retrieval Effectiveness Based on the Weak Ordering Action of Retrieval Systems"; American Documentation, 19 (1): 30–41, 1968.

[de Borda 1781] de Borda, J.: "Memoire sur les Elections au Scrutin"; 1781. Histoire de l'Academie Royale des Sciences, Paris.

[de Condorcet 1785] de Condorcet, M.: "Essai sur l'application de l'analyse a la probabilite des decisions rendues a la pluralite des voix"; 1785. Paris.

[Dean et al. 2002] Dean, M., Connolly, D., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L.: "OWL Web Ontology Language 1.0 Reference"; 2002. http://www.w3c.org/TR/owl-ref.

[Ding et al. 2005] Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., and Kolari, P.: "Finding and Ranking Knowledge on the Semantic Web"; In Procs of the 4th Intern Semantic Web Conference, 2005.

[FORTH-ICS 2005] FORTH-ICS: "The ics-forth rdfsuite: High-level scalable tools for the semantic web"; 2005. http://139.91.183.30:9090/RDF/.

[Freeman 1977] Freeman, L. C.: "A set of measures of centrality based on betweenness"; Sociometry, 40: 35–41, 1977.

[Freeman 1979] Freeman, L. C: "Centrality in social networks: Conceptual clarification"; Social Networks, 1: 215–239, 1979.

[Hayes 2004] Hayes, P.: "RDF semantics,w3c recommendation"; February 2004. http://www.w3.org/TR/rdf-mt/.

[Kemeny 1959] Kemeny, J.: "Mathematics without Numbers"; Daedalus, 88: 571–591, 1959.

[Motwani and Raghavan 1995] Motwani, R., and Raghavan, P.: "Randomized algorithms"; Cambridge University Press, 1995.

[Noy et al. 2001] Noy, N. F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R. W., and Musen, M. A.: "Creating semantic web contents with protege-2000"; IEEE Intelligent Systems, 2 (16): 60–71, 2001.

[Patel et al. 2003] Patel, C., Supekar, K., Lee, Y., and Park, E.: "OntoKhoj: a semantic web portal for ontology searching, ranking and classification"; Procs of the 5th ACM international workshop on Web information and data management, pages 58–61, 2003.

[Robertson 2007] Robertson, S.: "On Score Distributions and Relevance"; In Procs of the 29th European Conference on Information Retrieval, ECIR'2007, Rome, Italy, April 2007.

[Rosch 1978] Rosch, E.: "Principles of Categorization"; pages 27–48. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1978.

[Sabou et al. 2006] Sabou, M., Lopez, V., Motta, E., and Uren, V.: "Ontology Selection: Ontology Evaluation on the Real Semantic Web"; In Procs of 4th Intern. EON Workshop, Evaluation of Ontologies for the Web (colocated with WWW'2006), Edinburgh,United Kingdom, May 2006.

[Sheth et al. 2005] Sheth, A. , Aleman-Meza, B. , Arpinar, I., Ramakrishanan, C., Halaschek, C., Anyanwu, K., and Kochut, K.: "Semantic Association Identification and Knowledge Discovery for National Security Applications"; Journal of Database Management, 16 (1): 33–53, 2005.

[Theoharis 2007] Theoharis, Y: "On Power Laws and the Semantic Web"; Master's thesis, Computer Science Department, University of Crete, February 2007. http://athena.ics.forth.gr:9090/RDF/publications/MasterThesisTheohari.pdf

[Tzitzikas 2001] Tzitzikas, Y.: "Democratic Data Fusion for Information Retrieval Mediators"; In ACS/IEEE International Conference on Computer Systems and Applications, Beirut, Lebanon, June 2001.

[Tzitzikas and Hainaut 2005] Tzitzikas, Y., and Hainaut, J.-L.: "How to Tame a Very Large ER Diagram (using Link Analysis and Force-Directed Placement Algorithms)"; In Proceedings of 24th Int. Conf. on Conceptual Modeling, ER'2005, Klagenfurt, Austria, October 2005.

[van Ham et al. 2004] van Ham, F., van Wijk, J., and Eindhoven, T.: "Interactive Visualization of Small World Graphs"; Information Visualization, 2004. INFOVIS 2004, pages 199–206, 2004.

[Zhuge and Zheng 2003] Zhuge, H., and Zheng, L.: "Ranking Semantic-linked Network"; In Procs of the 12th International World Wide Web Conference, 2003.