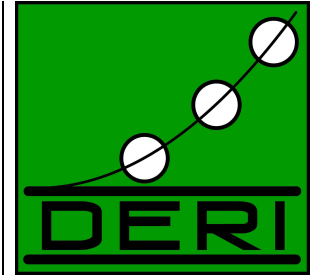


DERI – DIGITAL ENTERPRISE RESEARCH INSTITUTE



TOPDIS: TENSOR-BASED
RANKING FOR DATA SEARCH AND
NAVIGATION

DERI TECHNICAL REPORT 2009-06-21
JUNE 2009

Copyright © 2009 by the authors.

DERI – DIGITAL ENTERPRISE RESEARCH INSTITUTE

DERI Galway
IDA Business Park
Galway, Ireland
www.deri.ie

TOPDIS: TENSOR-BASED RANKING FOR DATA SEARCH AND
NAVIGATION

Andreas Harth Sheila Kinsella

Abstract. Web sources increasingly use the Resource Description Format (RDF) as a means for general-purpose knowledge representation. Many database-backed sites are now being complemented with a structured representation of their content; for example, the content of blogs, social networking sites, and Wikipedia is being made available in RDF and can be crawled and aggregated. Searching over large corpora of structured information collected from the Web brings about new challenges for ranking. One particular problem is that searches can result in graphs with thousands of edges, too large for a user to easily absorb. To this end, we introduce TOPDIS, a set of algorithmic tools to determine prominent elements in large semantic graphs. We provide a formalisation of the method using the concepts of multilinear algebra, evaluate scalability of the algorithm and quality of the results, and show how TOPDIS can improve search over structured data in general.

Acknowledgements: This work has been supported by Science Foundation Ireland project Lion (SFI/02/CE1/I131)

Copyright © 2009 by the authors

1 Introduction

The availability of massive amounts of structured data on the Web, published in a standard format, has the potential of enabling new ways of information seeking and browsing. Increasingly Web sources use the Resource Description Format (RDF) to publish a structured representation of their content. Available sources includes large monolithic datasets such as DBLP (Computer Science publications), Swissprot (Protein database), dbpedia (Wikipedia in RDF), TAP (general-purpose knowledge base), DMOZ (web page taxonomy), Geonames (geographic information), as well as a large number of small data files in various vocabularies dispersed across the Web. Data publishers, naturally, may have different world-views and diverse backgrounds, which yields a motley assortment of identifiers for instances, properties, and classes. Although a few standard vocabularies (such as FOAF for people, RSS/Atom and SIOC for online communities, and SKOS for classification schemes) have begun to emerge through social processes, the Web information space assembled by millions of people is still immensely diverse. Dealing with the breadth of data published online remains a unique challenge, especially for engineers who have to design systems and algorithms that are unaware of the schemas used in the data. One question pertains to prioritising elements of semantic data graphs automatically, either because manual assignment of weights to thousands of elements is impractical, or the schema is unknown a priori. In other words: how to bring order to the largely incongruous contributions of an enormous number of people?

Presenting only prominent or relevant alternative choices to the user is important since short-term memory in human brains is of limited capacity. Because web data is not organised in a predictable structure, often there are many choices on which terms to use in a query, or on which semantic links to navigate along in the information space. Therefore, an entity-based search engine for structured data should provide guided query construction where choices presented to the users are arranged based on their relevance. Given the power of distributed and loosely coordinated modelling of data, typically entities are described across a number of Web sources and thus the amount of data pertaining to one entity may become large. The result for a specified search term or query may be in the order of hundreds or thousands of statements – far too many for a user to quickly comprehend. Thus, a results page should show only the relevant or most prominent portion of the answer. Both means reduce the cognitive load of the user and thus allow for enhanced information take-up.

The problem we address is difficult because of three characteristics of Web data: domain independence, variance, and scale. Data on the Web can cover as disparate domains as social networks (e.g. from social networking sites), encyclopedic knowledge (e.g. Wikipedia), and scientific datasets (e.g. scientific literature, protein databases). When the dataset under examination is known a priori, researchers have the comfort of manually selecting certain properties of the dataset and developing specialised algorithms that take the background knowledge of a domain expert into account. In contrast, data on the Web is just too diverse to devise specialised algorithms for each dataset published since there are just too many. Data on the Web is not very uniform; entities may be described in hundreds of statements, or in hundreds of thousands. Dealing with that variance is challenging. Last but not least the Web brings about issues of scale. Thus, we made sure our method can be applied to very large datasets.

The problem we address is quite recent and unique to data integration on the Web. The architecture of the Web allows us to reach new dimensions in terms of scale and complexity of data integration, which in turn brings about new challenges. Previous approaches for ranking on the

Web operated on documents; web search engines typically include algorithms operating on the link graph such as PageRank [22] and HITS [13] to sort documents according to relevance. However, the methods developed for the document link graph do not leverage the full ranking possibilities on data graphs, which typically add the ability to use typed links between nodes. Previous approaches for ranking structured data were applied in the database area, where mediated schemas are hand-crafted and relatively modest in size; thus, assigning weights to different link types is still feasible (e.g. see [12]). Another aspect quite new to large scale data integration is that of data provenance. Tracking the origin of pieces of information is important on the Web, where anybody can say anything about everything. We show how to include the notion of data sources and provenance into our procedure, which can be seen as a special form of a more general concept of context.

Our method operates on data encoded as directed labeled graphs with context, where context is used to track the origin of subgraphs in the aggregated graph. As a first step, we use connectivity analysis to derive ranks for all elements in the graph, that is, nodes and labeled edges, and the sources of the data. The intuition behind our approach is that all elements in the directed labeled graph mutually influence each other, and the algorithm uncovers the implicit relations and assumptions that are represented by cumulative human input that created the data. We use multilinear algebra as a mathematical tool, in particular a structure called tensor, which is a generalisation of vectors and matrices to the n -dimensional case. We adapt the Power method for the tensor model to derive element-level ranks, and evaluate various methods of combining the element-level ranks into combined scores for combinations of elements. Please note that although a large number of recent work uses multilinear algebra for knowledge discovery and data mining [17] [26] [25] [27], these methods use a domain-specific encoding of the underlying data (dimensions are “customers”, “products”, “authors”, “keywords”, and so on), while our approach encodes the RDF with context data model into a tensor of four dimensions (subject-predicate-object-context). The work of Kolda and Bader [14] inspired this study and our method is a natural extension to the four-dimensional tensor case, however, we operate on a RDF data graph rather than a HTML document structure, provide means to aggregate individual element ranks, and our system scales to datasets which are orders of magnitude larger.

The contributions of this note are as follows:

- We present an algorithm based on a fixpoint calculation to automatically determine rankings for each element in quadruples (TOP method).
- We show the correspondence between a semantic graph with context and a four-dimensional tensor, and describe a space-efficient data structure to encode the four-dimensional tensor.
- We present a set of methods for combining the individual ranks into a compound rank score for quadruples or parts thereof (DIS method).
- We provide an experimental evaluation on a large RDF dataset consisting of over 200k RDF files collected from the Web, including a data representation of Wikipedia.

The remainder of the paper is organised as follows: Section 2 introduces preliminaries, a semantic search scenario, and one example graph from real Web data, together with ranking results. Section 3 provides an overview of the approach, Section 4 and 5 describe the TOP and DIS algorithm, respectively. Section 6 provides an experimental runtime performance and quality evaluation on a large dataset collected from the Web. Section 7 compares our approach with related work, and Section 8 concludes.

2 Semantic Search/Navigation

In the following we define triples, the atomic unit of information in RDF, and context, an extension to be able to track the origin of triples. And, we provide a motivating scenario where our ranking and rank aggregation method applies.

2.1 Triples and Context

The standard RDF data model is described in various W3C Recommendations, e.g. see [18]. For convenience and to save space, we use the namespaces in Table 1 to abbreviate URIs, analogue to Notation3¹ syntax. We use the term entity to refer to any real-world thing denoted online by a URI, such as a person, a protein, a topic or a location. For example, the entity Tim Berners-Lee is identified by the URI `http://www.w3.org/People/Berners-Lee/card#i`, abbreviated as `timbl:i`.

Prefix	Namespace
<code>rdf</code>	<code>http://www.w3.org/1999/02/22-rdf-syntax-ns#</code>
<code>rdfs</code>	<code>http://www.w3.org/2000/01/rdf-schema#</code>
<code>foaf</code>	<code>http://xmlns.com/foaf/0.1/</code>
<code>pim</code>	<code>http://www.w3.org/2000/10/swap/pim/contact#</code>
<code>timbl</code>	<code>http://www.w3.org/People/Berners-Lee/card#</code>
<code>aifb</code>	<code>http://www.aifb.uni-karlsruhe.de/Personen/viewPersonOWL/</code>

Table 1: Namespace abbreviations

(RDF Triple, RDF Node) Given a set of URI references \mathcal{R} , a set of blank nodes \mathcal{B} , and a set of literals \mathcal{L} , a triple $(s, p, o) \in (\mathcal{R} \cup \mathcal{B}) \times \mathcal{R} \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{L})$ is called an RDF triple,

In such a triple, s is called the subject, p the predicate, and o the object. An example for a triple is `timbl:i foaf:homepage <http://www.w3.org/People/Berners-Lee>`. The triple states Tim's homepage.

Although the RDF specification itself does not define the notion of context [10], usually applications require context to store various kinds of metadata for a given set of RDF statements.

(Triple in Context) A pair (t, c) with t be a triple and $c \in (\mathcal{R} \cup \mathcal{B})$ is called a triple in context c .

A triple $((s, p, o), c)$ in context c is equivalent to the quad (s, p, o, c) . The interpretation of context depends on the application. In our information integration use case, context denotes the URI of the file or repository from which a triple originated. Capturing provenance is one of the fundamental necessities in open distributed environments like the Web, where the quality of data has to be judged by its origin.

We use the term statement to refer to a triple or a quadruple, and call a part of a statement an element of the statement. Please note that the same triple can occur in multiple contexts. For example, all three data source in 1 contain the triple stating Tim's homepage.

¹<http://www.w3.org/DesignIssues/Notation3.html>

2.2 Search Engine Scenario

Consider a system aggregating a large chunk of the available structured data on the Web. The search engine has the following functionality for interrogating the aggregated data graph:

- Presenting search results. Given a set of keywords (e.g. `tim berners lee`), the system returns the entities matching the keyword. Similar to web search engines, the resources matching specified keywords should be prioritised according to relevance.
- Displaying entity information. A user who has identified an entity of interest (e.g. `timbl:i`) can request a page containing all information pertaining to the entity. The engine might return hundreds or even thousands of statements describing the entity, far too many for a user to quickly comprehend. Thus, the entity page should only contain the top-n relevant statements.

In the following we illustrate the problem of deciding on the most relevant top-n quadruples for a given entity, in this case Tim Berners-Lee. The problem is different to typical graph-based ranking scenarios because we require ranks for entire quadruples rather than ranks for individual nodes. The data presented here is a simplified version of the real dataset as collected from the Web. For the properties of the complete subgraph see Section 6.

Our example is based on a typical usage scenario of today's Semantic Web, involving social network data, expressed in the Friend of a Friend (FOAF)² vocabulary. Figure 1 shows parts of the different datagraphs obtained from three different sources which refer to the focus node. We show only a few statements here, however, the total number of quadruples describing Tim in our dataset is 197 from 28 different sources. In order to provide a concise characterisation of all the available information about Tim, we want to display only the most prominent statements. When integrating the different subgraphs, the provenance of information is an important aspect that needs to be taken into account in subsequent processing steps. To record the origin of a given triple, we use the notion of context which means that our data model consists of quadruples `{subject, predicate, object, context}`.

While most of the data in the example is expressed in FOAF, we cannot make any guarantees that this will always be the case. In particular, when collecting the files we also get data expressed in other vocabularies, which are not necessarily known a priori. Please also note that although we show a person in our example, all types of entities (such as proteins, locations, publications, etc.) could be returned, each of them described using completely different vocabularies, which renders manual intervention infeasible. The main point to observe here is that our algorithm is agnostic to the vocabularies used, and finds the most important elements in an unsupervised manner.

3 Overview of TOPDIS

The TOPDIS method utilises a link analysis algorithm (called TOP) to calculate importance scores for each element in a graph. A second step called DIS takes these scores as input to an aggregation method to rank each statement according to the importance of its constituent elements. Finally, the top-k statements are selected heuristically to derive a condensed view of the input graph.

²<http://www.foaf-project.org/>

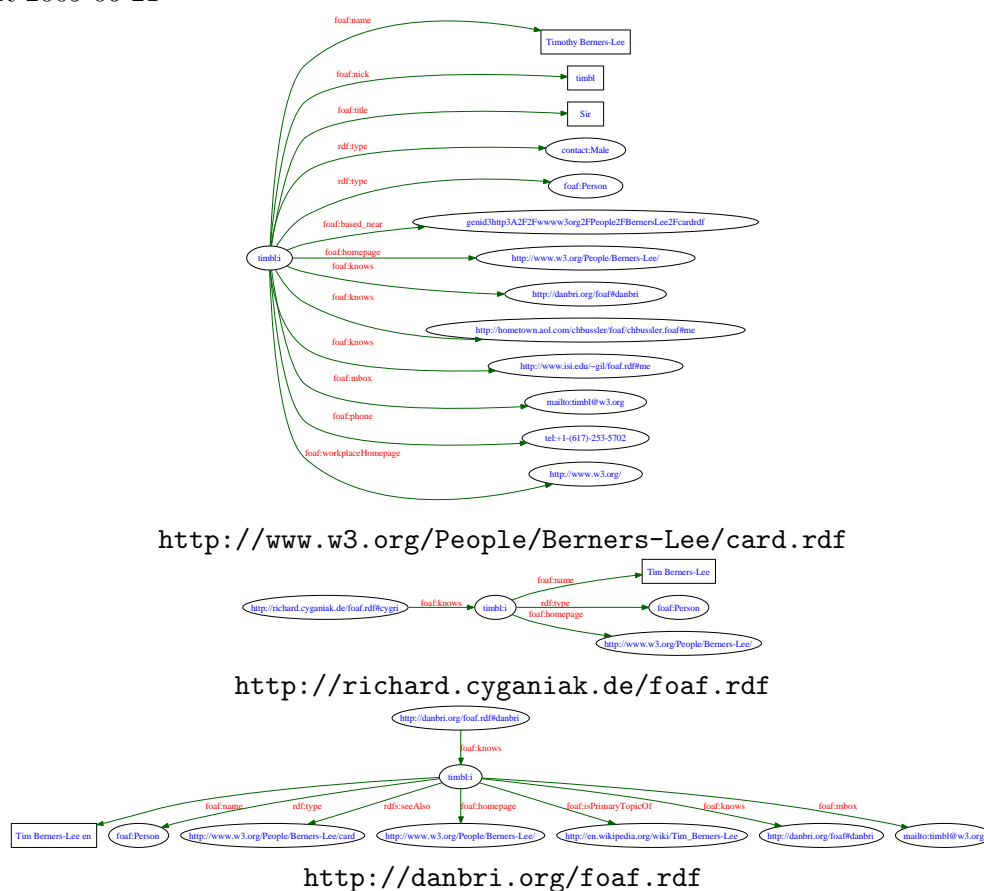


Figure 1: Three sources mentioning the focus node URI. Each data source represents one context.

The aim of the method presented here is to identify the most relevant information in a graph and condense it into one coherent representation. Firstly, we may need to identify the part of the graph pertinent to our query, if there is one. We then assess the importance of each subject, predicate, object and context using a connectivity analysis algorithm that takes into account different but interdependent properties of the graph. Finally, we combine these element-level rankings into statement-level rankings to prioritise the statements and extract the most important ones to derive a condensed view of the input graph.

In other words, the method consists of the following steps:

- Select either the entire graph as input to the algorithm, or extract a subgraph with all the nodes in the vicinity of selected focus nodes.
- Calculate rank scores for each element (subject, predicate, object, context) in the selected graph using a link-based connectivity analysis method (TOP algorithm).
- Determine the statements-level scores for each element in the subgraph by combining the rank scores for elements into a statement score (DIS algorithm).
- Apply a heuristic to select a representative subset of the most relevant statements.

3.1 Selecting the Subgraph

We can influence the outcome of the algorithm by choosing different strategies for deriving the input graph, in particular varying a parameter ϵ that determines the breadth of the selection. We can either use a small graph, which favours the central nodes ($\epsilon = 1$), or a larger graph (e.g., $\epsilon = 2$) with greater distance to derive more general rankings. Figure 2 shows graphically the queries used to select a subgraph with $e = 1$ and $e = 2$ from the focus node. When selecting a larger graph we can see the effects of topic drift – the method might highly rank elements that are not relevant to the initial query. The most general ranking is derived by selecting the entire graph as input to the TOP algorithm, thus computing a global rank of each element in the graph independent of any query. See Section 6 for experimental results on graphs with varying neighbourhood selection size.

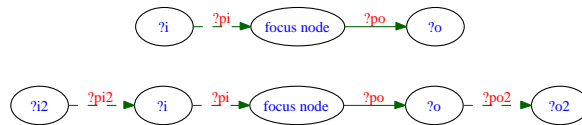


Figure 2: Queries used to select subgraphs with $\epsilon = 1$ and $\epsilon = 2$ (dashed lines denote optional matches).

In the selected subgraphs, we ignore statements containing literals which can be seen as attributes of a node rather than part of the graph topology. Also, we discard self-referential statements that would otherwise accumulate overly high scores. In addition, we do not consider `rdf:type` and `owl:sameAs` statements, since these schema-level statements deal with the representation of the data rather than the instances themselves.

3.2 Computing Element Ranks

The aim of the method presented here is to identify the most relevant information in a graph and optionally condense it into one coherent representation. First, we assess the importance of each subject, predicate, object and context using a connectivity analysis algorithm that takes into account different but interdependent properties of the graph. The TOPDIS method utilises a link analysis algorithm (called TOP) to calculate importance scores for each element in a graph. The results of the element ranking for the network shown in Figure 1 are provided in Table 2.

According to our element rankings, the most important subject within hop 1 of Tim Berners-Lee is `timbl:i`, as expected. The next highest ranked subjects are two documents created by Tim Berners-Lee, followed by two events in which he participated. In the object rankings, four of the top five positions are occupied by entities representing people, with `timbl:i` again ranking highest. The only other object entity ranked within the top five is Tim Berners-Lee’s homepage (<http://www.w3.org/People/Berners-Lee/>). All of these entities are indeed relevant and important considering the query at hand.

The highest ranking predicate in the graph is calculated as `foaf:knows`, which accounts for the most essential connections in the network. Other important predicates in the network include `rdfs:seeAlso`, which is used to indicate a resource providing additional information relating to the subject resource, properties describing personal information from the FOAF vocabulary (e.g. `foaf:homepage`), and SWAP Personal Information Markup predicates (`pim:participant`).

Subjects	Score
timbl:i	627432
http://dig.csail.mit.edu/breadcrumbs/blog/4	356325
http://dig.csail.mit.edu/2007/01/camp/data#course	356325
http://www.ecs.soton.ac.uk/~dt2/dlstuff/www2006_data#panel-panelk01	353858
http://wiki.ontoworld.org/index.php/_IRW2006	353858
Predicates	Score
foaf:knows	619492
foaf:homepage	424084
rdfs:seeAlso	422884
foaf:maker	407767
pim:participant	392859
Objects	Score
timbl:i	166413
http://www.w3.org/People/Berners-Lee/	153487
timbl:amy	144570
http://www.w3.org/People/karl/karl-foaf.xrdf#me	135552
http://www.w3.org/People/EM/contact#me	135552
Context	Score
http://www.w3.org/People/Berners-Lee/card	587689
http://danbri.org/foaf.rdf	401159
http://kuruman.org/foaf.rdf	375509
http://richard.cyganiak.de/foaf.rdf	349754
http://www.w3.org/People/EM/contact	342809

Table 2: Results from ranking the graphs represented in Figure 1

The context which ranks first is Tim’s machine readable homepage file, the main source of the information about Tim.

3.3 Combining Element Rank Scores

Having computed the four sets of ranks, we can combine the element ranks to derive at a compound ranking score according to the importance of its constituent elements, necessary to e.g. display the most prominent triples or provide navigation cues. In addition to calculating combined rank scores, we provide a heuristic to select a representation sample of the most popular statements. Figure 3 shows the final result of the TOPDIS algorithm in combination with a heuristic: a distilled graph derived from the input graph in Figure 1. Please note that a graphical representation is just one way of rendering the resulting graph, other views can also be generated, ideally taking into account styling information that specifies how certain properties should be displayed.

4 TOP: Computing Element Ranks

In the following section we discuss in detail how to derive element ranks using the TOP algorithm. The TOP algorithm is a link-based connectivity analysis method in the tradition of PageRank and



Figure 3: Distilled graph containing top-10 statements.

HITS (and, more recently, TOPHITS), applied to four-dimensional data. To formally describe our algorithm, we introduce the notion of a tensor, a mathematical structure used in multilinear algebra. A tensor is a generalisation of vectors and matrices to the n -dimensional case. That is, a vector can be seen as a tensor of rank 1 and a matrix as a tensor of rank 2. The four-dimensional model of directed labeled graphs with context can be captured by a four-way tensor.

4.1 Calculating Importance Scores

The intuition behind the TOP method is that the importance scores of each of the elements in a statement mutually influence each other. Important subjects point to important objects, and important objects are pointed to by important subjects. Important predicates link important subjects and objects. Important triples occur in important contexts. By taking into account these mutual associations, we aim to distil a comprehensible outline of the most significant features from the aggregated datagraph.

Importance scores are calculated as follows:

$$\mathbf{s}_i^{t+1} = \sum_{i \xrightarrow{k} j @ l} \mathbf{p}_k^t \mathbf{o}_j^t \mathbf{c}_l^t \text{ for } i = 1, \dots, n \quad (1)$$

$$\mathbf{p}_j^{t+1} = \sum_{i \xrightarrow{k} j @ l} \mathbf{s}_i^{t+1} \mathbf{o}_j^t \mathbf{c}_l^t \text{ for } j = 1, \dots, n \quad (2)$$

$$\mathbf{o}_k^{t+1} = \sum_{i \xrightarrow{k} j @ l} \mathbf{s}_i^{t+1} \mathbf{p}_k^{t+1} \mathbf{c}_l^t \text{ for } k = 1, \dots, n \quad (3)$$

$$\mathbf{c}_l^{t+1} = \sum_{i \xrightarrow{k} j @ l} \mathbf{s}_i^{t+1} \mathbf{p}_k^{t+1} \mathbf{o}_j^{t+1} \text{ for } l = 1, \dots, n. \quad (4)$$

where $i \xrightarrow{k} j @ l$ means there is a link from node i to node j with predicate k and context l . The subject score is updated with the product of the predicate, object, and context scores of the particular statement. The predicate score is updated with the product of the subject, object, and context scores. The object score is updated with the product of the subject, predicate, and context

scores pertaining to the statement. The context score is updated with the product of the subject, predicate, and object scores.

During every iteration, after calculating each result vector a , we normalise so that the minimum element score is 1 and perform the scaling

$$\mathbf{a}_i = 1 + \log(\mathbf{a}_i) \text{ for } i = 1, \dots, n \quad (5)$$

to prevent a few prominent elements from dominating the results.

4.2 Storing the Tensor

Our algorithm requires sequential access to each of the dimensions of the tensor. Naively encoding a four-way tensor with dimensions i, j, k, l would require space of size $i * j * k * l$, which is clearly too expensive for any application to real-world problems. For an efficient representation of sparse tensors we devise a data structure that allows access to all statements pertaining to a given subject, predicate, object, or context via a sequential scan. For grouping together statements with given elements, we generate four index files which are sorted according to each of the four dimensions. We use a disk-based data structure to be able to handle large tensors, but the disk-based structure can easily be replaced by a similar in-memory data structure. The index files are created via multiway merge sort with a complexity of $O(n \log n)$, and compressed using Huffman coding to save disk space.

Figure 4 illustrates the index structure. The index file is sorted according to subject, which means we can access all predicate, object, and context nodes by just scanning sequentially through the index files. For the purpose of the algorithm, we employ four index files, each one grouped according to a different dimension. For smaller input files, we also devised an equivalent in-memory data structure.

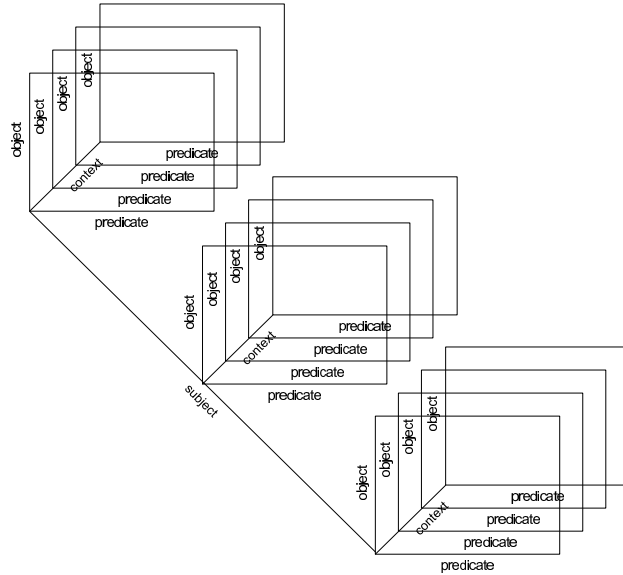


Figure 4: Tensor index with four dimensions subject, predicate, object, and context.

4.3 Computing Ranks Iteratively

Once the input tensor is selected and the index files prepared, we apply an iterative algorithm (Algorithm 1) to compute individual TOP scores. In essence, we use an adaptation of the Power Method [9] to the multidimensional case. First, the element vectors are initialised to 1. Next, for each of the subject, predicate, object, and context vectors, we update the element rank vector in terms of the rank vectors of the other elements in a statement, and normalise and scale the vectors.

Algorithm 1 The TOP algorithm for deriving ranks from graph topology on directed labeled graphs with context.

Require: Graph G encoded in subject, predicate, object, context indices

$\mathbf{s} \leftarrow 1, \mathbf{p} \leftarrow 1, \mathbf{o} \leftarrow 1, \mathbf{c} \leftarrow 1$

while error too big or number of iterations lower than threshold **do**

for all quadruple $sp_k o_j c_l$ with subject $s \in G$ **do**

if repeating s **then**

$\mathbf{s}[s] \leftarrow \mathbf{s}[s] + \mathbf{p}[p_k] * \mathbf{o}[o_j] * \mathbf{c}[c_l]$

else

$\mathbf{s}[s] \leftarrow \mathbf{p}[p_k] * \mathbf{o}[o_j] * \mathbf{c}[c_l]$

end if

end for

 normalise and scale \mathbf{s}

 calculate \mathbf{o}, \mathbf{p} , and \mathbf{c} correspondingly

end while

return element ranks $\mathbf{s}, \mathbf{p}, \mathbf{o}, \mathbf{c}$

We assume that our algorithm has sorted access to all statements with a given subject, predicate, object, or context using a multidimensional access structure to be able to iterate over each of the four dimensions and carry out the calculation. There are many possible implementations of this multidimensional access structure; we have implemented both disk-based and in-memory access structures.

We have found that directly applying the calculations without a scaling operation results in a winner-takes-all phenomenon, where a few closely-knit elements accumulate all rank in the system and dominate the results, regardless of their relevance.

The complexity of the algorithm is linear if we assume sorted access enabling the rank calculation along a particular dimension via an index scan. The running time of the algorithm is mainly determined by the number of iterations needed to reach a fixpoint.

4.4 Mathematical Characterisation

In this section we describe our method in terms of multilinear algebra. We use the tensor notation as described in [19]. Let \mathbf{A} be a four-mode tensor $I_1 \times I_2 \times I_3 \times I_4$ corresponding to subject, predicate, object and context dimensions. The vectors $\mathbf{s}, \mathbf{p}, \mathbf{o}$ and \mathbf{c} are updated during each iteration by

$$\mathbf{s}^{t+1} = \mathbf{A} \times_2 \mathbf{p}^t \times_3 \mathbf{o}^t \times_4 \mathbf{c}^t \quad (6)$$

$$\mathbf{p}^{t+1} = \mathbf{A} \times_1 \mathbf{s}^{t+1} \times_3 \mathbf{o}^t \times_4 \mathbf{c}^t \quad (7)$$

$$\mathbf{o}^{t+1} = \mathbf{A} \times_1 \mathbf{s}^{t+1} \times_2 \mathbf{p}^{t+1} \times_4 \mathbf{c}^t \quad (8)$$

$$\mathbf{c}^{t+1} = \mathbf{A} \times_1 \mathbf{s}^{t+1} \times_2 \mathbf{p}^{t+1} \times_3 \mathbf{o}^{t+1} \quad (9)$$

5 DIS: Combining Rank Scores

The ranking scores obtained by the TOP method allow to prioritise individual elements (subjects, predicates, objects, and context). However, applications require to prioritise not only single elements, but pairs, triples, or even quadruples. For example, displaying only the top-k triples pertaining to an entity requires to prioritise triples, not just single elements. In this section, we describe a rank score aggregation step termed DIS which allows to combine individual rank scores into an aggregated rank score.

5.1 Vector Norm

Combinations of n elements can be interpreted as a n -vector. For example, pairs such as subject/object combinations or predicate/object combinations are treated as 2-vector, and triples such as subject/predicate/object are treated as 3-vectors, and so on. We use the length of the vector to calculate the combined rank.

The overall rank distillation procedure is then for quadruples:

$$\|x\| = \sqrt{s^2 + p^2 + o^2 + c^2} \quad (10)$$

For other element combinations the distilled ranked is calculated analogously.

5.2 Picking Representative Statements

There is one final step missing to arrive at a subgraph containing only the top-k most representative triples. A naive way would be to just select the top-n statements after calculating the vector norm. However, given uniform input data, typically one or two highly popular predicates dominate the top ranks. As a result, the top-k statements might all include the same predicate, which typically does return very uniform subgraphs which not give a just characterisation of the focus node. To overcome this behavior, we devise an algorithm that picks the top-k statements while maintaining some diversity in the results returned. The heuristics for deriving distilled graphs for element combinations is shown in Algorithm 2.

6 Experiments and Evaluation

We now present the results of a study to validate the methods and algorithms described in this paper. We begin by characterising the dataset obtained from the Web, then describe the experimental setup and performance characteristics of the method, and finally introduce the quality evaluation method and results. We use a set of baseline algorithms for comparison for our TOP and DIS steps.

Algorithm 2 Heuristics for deriving distilled graphs.

Require: Graph G

Require: s, p, o, c with element rank scores

Require: Compound rank mask m

Require: Threshold k

for each quadruple $spoc$ **do**

$c =$ compound rank for $spoc$

end for

sort c

for each entry e in c AND less than k results printed **do**

if there exists multiple element combinations with same rank value **then**

 pick one at random and print element combination and rank value

else

 print element combination and rank value

end if

end for

We compare TOP with a frequency count (FREQ) in the performance and quality evaluation, and HITS to cross-check subject and object scores, and to compare runtime performance. We compare the quality of the DIS step with summation (SUM) and multiplication (MUL) of element scores.

6.1 Dataset

We collected a dataset with 72,462,443 statements from 222,469 RDF sources using MultiCrawler [11]. Large database-backed sites³ were only partially included in the dataset. The dataset contains a wide variety of entities (27,399 different classes) and connections between them (107,487 distinct predicates). The size of the uncompressed data file is 17 GB, the sorted and compressed indices are between 1.8 and 2.0 GB. The difference in size for the index files can be attributed to different compression ratios depending on the sorting order.

6.2 Implementation

We have implemented a prototype of the TOPDIS algorithm in Java. The experiments were carried out on AMD Opteron 2.2GHz machines with 4 GB of main memory running Debian Linux.

For the experiments on the TOP step we fixed the number of iterations. Typically, ten iterations are enough to return reasonably stable result vectors. Fixing the number of iterations has the added benefit that we do not need to store the results of the previous iterations in memory, thus halving the amount of main memory needed. We use four in-memory hashtables to hold the result vectors obtained during each iteration.

For the DIS step, we sequentially read all statements, compute the final score for each, and keep the resulting top- k statements in memory.

³such as livejournal.com, tribe.net or vox.com

6.3 Performance Results

For the performance evaluation we derived two sets of subgraphs. Table 3 list the small subgraphs pertaining to an entity (e.g. `timbl:i`), and Table 4 the broad subgraphs containing entities of a given type (e.g. all entities of `rdf:type foaf:Person`) to demonstrate the scalability of the algorithms. For the subgraphs describing an entity, we select two sets of subgraphs with $\epsilon = 1$ and $\epsilon = 2$ to show the effects of choosing subgraphs of varying sizes on the runing time of the algorithm.

URI	ϵ	No of stmt	TOP	HITS	FREQ
<code>dbpedia:Beijing</code>	1	393	536	432	339
<code>dbpedia:Beijing</code>	2	1383	1212	786	715
<code>dbpedia:DNA</code>	1	41	317	58	39
<code>dbpedia:DNA</code>	2	50	262	62	43
<code>dbpedia:Galway</code>	1	79	332	90	49
<code>dbpedia:Galway</code>	2	322	700	384	317
<code>dbpedia:Republic_of_Ireland</code>	1	916	842	748	518
<code>dbpedia:Republic_of_Ireland</code>	2	5008	2856	1624	1196
<code>aifb:id57instance</code>	1	1085	869	668	513
<code>aifb:id57instance</code>	2	8260	4479	2283	1347
<code>timbl:i</code>	1	197	490	329	71
<code>timbl:i</code>	2	3371	1709	1174	982

Table 3: Ranking times for various entity subgraphs (times in ms, ten iterations, in-memory index).

Restriction	No of stmt	Indexing time	TOP	HITS	FREQ
<code>rdf:type foaf:Person</code>	5201263	1738406	21077732	6885006	397181
<code>rdf:type skos:Concept</code>	62358	23694	41078	18587	6295

Table 4: Ranking times for subgraphs containing all entities per class (outlinks only, times in ms, ten iterations, on-disk index).

6.4 Quality Evaluation

To evaluate the quality of the method, we conducted a user study in which we asked participants to manually rate, on a Likert scale from 1 to 5, the importance of statements pertaining to four of the entity subgraphs. Nine people participated in the study, resulting in four to six manual ratings for each of the tested subgraphs. The aggregated ratings represented the preferred statements as expressed by the participant group.

We decided to use Kendall τ distance for comparing the alternative algorithmic approaches with the manual ratings. An second option was to use precision and recall, which, however, requires a binary choice of relevance, which is hard to provide for semistructured data. Table 5 lists the four subgraphs tested, together with Kendall's τ that measures the correspondance between to ranked lists (where 1 means full correspondance, 0 no correspondance, and -1 means one list is the reverse of the other).

URI	FREQ/SUM	FREQ/MUL	FREQ/DIS	TOP/SUM	TOP/MUL	TOP/DIS
<code>timbl:i</code>	0.51111	0.51111	0.51111	0.51111	0.51111	0.51111
<code>aifb:id57instance</code>	-0.066667	-0.066667	-0.066667	-0.066667	-0.066667	-0.066667
<code>dbpedia:Beijing</code>	0.022222	-0.28889	0.022222	0.15556	-0.33333	0.15556
<code>dbpedia:Galway</code>	0.15556	0.11111	0.066667	0.15556	0.11111	0.15556

Table 5: Kendall’s τ distance that measures the overlap of top-10 results between algorithmic and manual ranking.

6.5 Discussion

TOP/SUM and TOP/DIS perform best on the four selected subgraphs. However, the quality result for the subgraphs are vastly different in terms of τ distance. One possible explanation is in the nature of the subgraphs (and sources). The URI `timbl:i` is used in a range of data sources spanning many people and organisations. The resulting subgraph contains quite a few redundant triples stemming from the diversity of user contributions. In a way, data publishers vote implicitly for important statements and element by using their URIs. It seems that given enough diversity in user contributions, simple frequency counts are enough to determine popular elements and statements.

In contrast, the URI `aifb:id57instance` that denotes Rudi Studer is not shared across many sites. The URI occurs in many contexts, however, most (if not all) of these seem to be database-generated from a central repository. There is not much human input and thus not much variance in the composition of the subgraph, which seems to counter the application of our method.

The dbpedia subgraphs appear to be suited for our ranking procedure, however, given the triples stem from one source, the results are not as good as for `timbl:i`.

During experimentation, we encountered link spam, especially with data converted from the HTML web. Using host names rather than full URIs for the context alleviates that particular problem, since link farms then do not acquire too much rank scores. Although this simple method worked on our dataset, more sophisticated spamming methods will very likely require more adaptations.

7 Related Work

Citation-based algorithms have been investigated in sociology in the area of social network analysis [24], which states as unresolved issue the mismatch between two-dimensional graph theory and multi-dimensional social networks.

We have extended links-based connectivity algorithms such as PageRank [22], HITS [13], and TOPHITS [14] to operate on data graphs. While PageRank scales very well, it only operates on two-dimensional matrices. HITS also operates on two-way matrices, however it derives two result vectors that correspond to hub scores and authority scores, and applies a process similar to Singular Value Decomposition to derive more than one set of results vectors (essentially calculating not only the dominant eigenvalue/eigenvector pair, but all eigenvalues/eigenvectors), denoting multiple communities or subgraphs. Similarly, TOPHITS uses as mathematical model the PARAFAC decomposition, which is an extension of SVD to the multi-dimensional case. In contrast to TOPHITS,

which works on a representation of documents, our approach takes in four-dimensional data, encoded as quadruples, taking into account the provenance of statements.

Bharat and Henzinger [4] discuss various problems of HITS such as nepotism and topic drift, and propose methods to fix these issues. An alternative algorithm, SALSA[16], performs random walks on web pages, a method which avoids the problem caused by tightly-knit communities. Authorities scores are defined by a random walk following a backward-link and then forward link alternately, and hub scores are defined by a random walk following a forward-link and then backward-link alternately. The HITS algorithm has been found to be unstable under some circumstances[20] and this problem has been addressed by a random surfer model in Randomised HITS[21], [23], and Randomised SALSA[15]. For a good overview of these connectivity-based ranking methods see [6].

For our application scenario, we only require the first set of vectors, in contrast to TOPHITS which computes all tensor decompositions. Latent semantic indexing [7] uses similar methods to HITS, also on a two-dimensional level, to analyse relationships between terms and documents.

We have demonstrated that our algorithm scales to large data sets, while most work on multi-dimensional representation so far has only focused on tensors of modest size. TOPDIS could utilise a two-level algorithm similar to [5] to combine global rankings with on-demand ranking for cutting down the size of intermediate results in a distributed search engine architecture.

ObjectRank [12] describes an approach to rank a directed labeled graph using PageRank. The work includes a concept called authority transfer schema graphs, which defines weightings for the transfer of propagation through different types of links. ObjectRank relies on user input to weight the connections between nodes to describe their semantic weight, so that the three-way representation can be collapsed into a two-way matrix, on which a PageRank-style algorithm is applied. Also, no consideration is given to the provenance of data.

SemRank [1] rank relations and paths on Semantic Web data using information-theoretic measures. In contrast, we rank all elements of a data graph with context, using a mathematical model rooted in multilinear algebra.

Rank aggregation methods in the realm of metasearch have been studied e.g. by [2] and [8]. Typically these methods assume to merge single lists of ranks into a combined list of single nodes. The Borda count is frequently used, however, for our approach we require more complex aggregation, since we have to derive a score for a combination of nodes, i.e. entire statements.

8 Conclusion

In this study, we have shown how to attach a value to RDF statements on the Web, and use that value to extract the essential elements of potentially very large graphs. We have applied methods from multilinear algebra to large datasets collected from thousands of sources. Our approach extends unsupervised link analysis algorithms known from hypertext applications to the realm of semantic graphs with context. We have given a mathematical characterisation of our algorithm, and verified experimentally the intuition behind the method. We apply the derived rankings to construct a distilled version of potentially large data graphs, arriving at a concise characterisation of the focus nodes. However, the algorithm can be applied to a wide range of other application areas involving graph-structured data.

Both TOP and DIS algorithms are rooted in a clean algebraic model and thus can easily be extended. For example, adding dimensions such as relevancy feedback scores, group membership or distance in a social network for personalised rankings is straightforward.

Uncovering the structure represented in collaboratively edited semantic graphs has significant scientific and commercial potential. Firstly, the Web, the largest artifact of human knowledge, becomes subject to scientific analysis [3]. Understanding the implicit connections and structure in the Web data graph can help to reveal new understandings of collaboration patterns and the processes by which networks form and evolve. Secondly, making sense out of scientific data published on the Web can help scientists to gain insights for their research. The ability to navigate and search effectively through a store of knowledge integrated from thousands of sources can broaden the pool of information and ideas available to a scientific community. Thirdly, making the Web data graph available for interactive querying, browsing, and navigation has applications in areas such as e-commerce and e-health, and has the potential to improve the quality and utility of Web search in general.

References

- [1] K. Anyanwu, A. Maduko, and A. Sheth. Semrank: ranking complex relationship search results on the semantic web. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 117–127, New York, NY, USA, 2005. ACM Press.
- [2] J. A. Aslam and M. Montague. Models for metasearch. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 276–284, New York, NY, USA, 2001. ACM Press.
- [3] T. Berners-Lee, W. Hall, J. Hendler, N. Shadbolt, and D. J. Weitzner. Creating a science of the web. *Science*, 313(11), 2006.
- [4] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, AU, 1998.
- [5] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 426–434, New York, NY, USA, 2003. ACM Press.
- [6] S. Chakrabarti. *Mining the Web*. Morgan Kaufmann Publishers, 2003.
- [7] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.
- [8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 613–622, New York, NY, USA, 2001. ACM Press.
- [9] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

- [10] R. V. Guha, R. McCool, and R. Fikes. Contexts for the Semantic Web. In *Proceedings of the 3rd International Semantic Web Conference, Hiroshima, Nov. 2004*.
- [11] A. Harth, J. Umbrich, and S. Decker. Multicrawler: A pipelined architecture for crawling and indexing semantic web data. In *5th International Semantic Web Conference, Athens, GA, USA., 2006*.
- [12] H. Hwang, V. Hristidis, and Y. Papakonstantinou. ObjectRank: a system for authority-based search on databases. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 796–798, New York, NY, USA, 2006. ACM Press.
- [13] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [14] T. Kolda, B. Bader, and J. Kenny. Higher-order web link analysis using multilinear algebra. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 242–249. IEEE Computer Society Washington, DC, USA, 2005.
- [15] H. Lee and A. Borodin. Perturbation of the hyperlinked environment. *Proceedings of the Ninth International Computing and Combinatorics Conference*, 2003.
- [16] R. Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Computer Networks*, 33(1-6):387–401, 2000.
- [17] M. W. Mahoney, M. Maggioni, and P. Drineas. Tensor-cur decompositions for tensor-based data. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 327–336, New York, NY, USA, 2006. ACM.
- [18] F. Manola and E. Miller. RDF Primer. W3C Recommendation, Feb. 2004. <http://www.w3.org/TR/rdf-primer/>.
- [19] C. D. M. Martin. Tenos decompositions workshop discussion notes, 2004.
- [20] A. Ng, A. Zheng, and M. Jordan. Link analysis, eigenvectors and stability. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 903–910, 2001.
- [21] A. Ng, A. Zheng, and M. Jordan. Stable algorithms for link analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 258–266. ACM Press New York, NY, USA, 2001.
- [22] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.
- [23] D. Rafiei and A. O. Mendelzon. What is this page known for? Computing web page reputations. *Computer Networks*, 33:823–835, 2000.
- [24] J. Scott. Trend report: Social network analysis. *Sociology*, 22(1):109–27, 1988.
- [25] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383, New York, NY, USA, 2006. ACM.

- [26] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 382–390, New York, NY, USA, 2005. ACM.
- [27] X. Yin, J. Han, and P. S. Yu. Linkclus: efficient clustering via heterogeneous semantic links. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 427–438. VLDB Endowment, 2006.