Title	ReConRank: A Scalable Ranking Method for Semantic Web Data with Context
Author(s)	Hogan, Aidan; Harth, Andreas; Decker, Stefan
Publication Date	2006
Publication Information	Aidan Hogan, Andreas Harth, Stefan Decker "ReConRank: A Scalable Ranking Method for Semantic Web Data with Context", Proceedings of Second International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006), in conjunction with International Semantic Web Conference (ISWC 2006), 2006.
Item record	http://hdl.handle.net/10379/492

ReConRank: A Scalable Ranking Method for Semantic Web Data with Context*

Aidan Hogan, Andreas Harth, and Stefan Decker

National University of Ireland, Galway
Digital Enterprise Research Institute
firstname.lastname@deri.org

Abstract. We present an approach that adapts the well-known PageRank/HITS algorithms to Semantic Web data. Our method combines ranks from the RDF graph with ranks from the context graph, i.e. data sources and their linkage. We present performance evaluation results based on a large RDF data set obtained from the Web.

1 Introduction

Ranking is an important component in most search engines to prioritise search results and to offer the user an immediate list of the most relevant results to a query. PageRank [8] has proven to be a useful and scalable algorithm to perform rankings on very large hypertext datasets.

As more RDF data is emerging online, a scalable, user-friendly search engine for Semantic Web data is becoming a more pertinent requirement. A Semantic Web search engine enables querying large RDF datasets, which requires ranking functionality to present results to the user in a meaningful way i.e. by prioritising relevant, important results.

Algorithms in the style of PageRank seem to be suitable candidates for ranking RDF knowledge bases, since RDF data forms inherent graphs. However, there are a number of differences between data coming from the Semantic Web and documents from the regular Web. This affects the methodology of such a links analysis ranking scheme.

1. In the HTML web, only the provenance of data is important. Very little meaningful structure can be derived from the data itself.
2. RDF data has varying types of links as opposed to HTML link structure which is based solely on hyperlinks. To paraphrase, whilst the HTML link structure is a directed graph, RDF is a directed labelled graph.
3. RDF resources may span various provenances or contexts. Thus information relating to a certain resource can be consolidated from multiple sources. The danger here is that anybody can make statements about anything, which can seriously affect the quality of the resulting data presented to the user.

* This work has been supported by Science Foundation Ireland (SFI/02/CE1/I131).

A ranking scheme for RDF data therefore has to take into account both the data graph and the data provenance graph. In this paper, we present ResourceRank, which can be used to perform ranking of RDF resources. We subsequently present a method to rank data sources or contexts, and finally reconcile the two methods with ReConRank.

One particular obstacle present to applying links analysis to RDF graphs collected from the Web is that the resources in the graphs might be not well connected or interlinked. In contrast, the contexts of RDF graphs are always linked, otherwise a crawler would not be able to locate the data. Since the quality of ranking depends heavily on how well the input graph is linked, adding context information to the graph can improve ranking quality considerably.

The ranking algorithm presented in this paper is tailored towards RDF data collected from the Web. For evaluation of the algorithm, outlined in Sect. 5, we use a large RDF data set consisting of 15M triples crawled from the Web from 70K different sources.

The contributions of this work are:

- We present ReConRank, an algorithm rooted in PageRank to rank resources in RDF files together with contexts to improve the ranking quality.
- We evaluate the performance of the implemented algorithm over a large dataset obtained from the Web.

The remainder of this paper is organised as follows: Section 2 presents an approach to select a topical subgraph which is used at a later stage for ranking. In Sect. 3 we present the basic algorithm including common optimisations to rank resources in RDF graphs. Section 4 extends the algorithm to include context information in the process. We describe results of performance tests in Sect. 5. Section 6 relates our approach with existing work, and Sect. 7 concludes.

2 ReConRank Environment

The intended deployment of the ReConRank methodology described herein, is as a ranking component for a Semantic Web Search Engine. Note, the algorithm is not specific to just this use-case, but easily adaptable to any architecture requiring RDF ranking. The search engine architecture is currently under development and outside of the scope of this paper. However a brief overview is necessary to clarify the intended application of the ReConRank algorithm.

The search engine obtains data through publicly available structured datasets and from the Web. The collected data is then indexed in an RDF store. On top of the index structure is a user interface component. The user can specify simple queries which the UI converts to queries for data in the index. The UI recieves and displays subgraphs which are returned as answers to the queries. Each resource in the subgraph is displayed as a separate result. Thus results are grouped by subjects within the RDF dataset.

Ranking is placed between the UI component and the index structure as a means of providing a more meaningful listing of results to the user. The requirements of the ranking component include that it be scalable and fast. The desired output includes metrics for ordering results (resources) and metrics for assigning trust values to information from different contexts.

The ranking component is a dynamic ranking system, and thus is invoked at query-time to rank data returned from the index structure. The algorithm only analyses results data: data that will ultimately be presented to the user, a different approach to PageRank's global application.

The original PageRank method [8] is applied to the entire graph before query-time. This approach has certain disadvantages. The entire data set might be excessively large; calculating ranking scores on such a large data set would require data structures that exceed the amount of memory available. There would be a large overhead regarding a distributed architecture for calculating PageRank. Also, upon indexing more data, the analysis would need to be re-computed. Thus regular computation would be necessary.

Another drawback of calculating a global ranking score is that universally popular pages (in terms of global PageRank) might not be the most relevant for a given query. For example, when searching for popular researchers in a given research area, e.g. Semantic Web, the ranking algorithm should return not just popular database researchers that have a cursory mention of Semantic Web somewhere, but researchers that are popular within the Semantic Web community.

We adopt a different approach similar to what is described by Kleinberg [7] as focused subgraphs. When the user specifies a query, say a simple keyword query, the indexing structure returns resources matching the keyword query and all data about those resources. Surrounding resources in the graph and their data may also be included. Then, we perform ranking only over the topical subgraph. The benefit of this approach is that relevancy is preserved. The result is a list of resources prioritised according to their popularity within the topical subgraph.

Again, a topical subgraph in our approach is selected as follows: given a literal matching a keyword, we select inlinks and outlinks of the subject node of the matching literal.

An important aspect affecting results is the size of the selected subgraph. We use a parameter n to specify how many hops in the vicinity of the matching literal should be included in the topical subgraph, that is, include nodes in the graph that are reachable in n steps from the subject node of the matching literal. Hops can be in either direction, through inlinks or outlinks. Figure 1 illustrates how to select a topical subgraph with a literal matching the keyword (bold) with $n = 1, 2, 3$. The value of n determines how broad or narrow a search result is.

One issue with the topical subgraph approach is that the resulting subgraph can be quite small. Intuitively links analysis works best on a reasonably well-linked dataset, and a small graph might be poorly interlinked. We present a means to overcome this obstacle in Sect. 4.

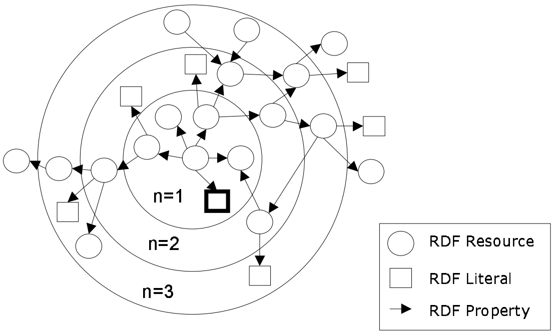

Fig. 1. Topical subgraph for matching keyword (bold) with $n = 1, 2, 3$. Selecting larger portions of the graph before ranking yields broader results (recall up, precision down), while selecting smaller portions yields narrower results (recall down, precision up).

The main challenge with the dynamic ranking approach is to make the ranking component fast enough to allow for ranking during query-time. We present our algorithm to rank resources in the following section and conclude that the performance challenge has been sufficiently met in Sect. 5.

3 Ranking Resources

In this section we review the basic PageRank algorithm which we implement, including common optimisations to speed up performance.

PageRank is an iterative computation which takes as input a graph consisting of nodes and edges. In each successive iteration, the score of node A is determined as a summation of the PageRank score in the previous iteration of all the nodes that link to node A divided by their number of outlinks. Traditionally, all nodes are initialised with an equal score before the first iteration; we adopt a different approach which is simply to initialise each node with the ratio of all links that the node receives as inlinks, which offers an enhanced initial guess with minimal overhead. In experimental evaluation, we found that this initialisation method reduced the number of iterations required by about one third.

Optional weightings functionality are provided within the algorithm, which acts as a parameter which can be manipulated to 'tweak' the algorithm using various techniques. Weightings is implemented in a matrix with identical structure and indexing to the connectivity matrix. Essentially, each entry in the matrix acts as a weight for the link in the corresponding index in the connectivity matrix. This functionality has not yet been used. For the most part we foresee the weightings as a means of allowing users leverage over the type of results that are prioritised by specifying various parameters, thus altering the weightings. When the ranking is incorporated into the larger search engine architecture, ranking parameters can be passed from the UI to bias the ranking towards a user preference.

For example, one possible application is to use TF-IDF [10] measures passed to the ranking engine to provide weightings. The TF-IDF measure is widely used to quantify relevancy through analysis of the prose of a document against a keyword query. Understanding the random surfer model as applied to PageRank[8], weightings can be derived from TF-IDF scores for resources with the justification that the surfer will follow a path through the graph attempting to stay on topic and is more likely to visit pages with high text relevancy. This approach is very similar to the ''intelligent surfer model' [9]. The exact nature of the effect of TF-IDF measures on the weightings could be specified by the user; perhaps as a slider bar adjusting priority to importance vs. relevance. This would have a significant effect in topical subgraphs where $n > 1$, keeping more relevant results near the top.

For some applications weightings may also apply to the graph. For instance, in some scenarios, it may be worth while to count the number of links present from one node in a graph to another and apply weightings to match. To justify with an example, author A co-authors four papers with author B and twelve papers with author C. Thereby, this is reflected in the ranking scheme where the link from author A to author C propagates three times the authority that the link to author B propagates.

One final application we foresee for weightings is outlined in Sect. 4. Ultimately, the weightings could be used to combine various preferences for ranking to improve the quality of the ranking, and provide the user with some leverage over the prioritisation of results.

Within the framework of the original PageRank algorithm is the damping factor, usually signified by d. It has a numeric value usually agreed upon as 0.85[8]. A node's voting power is in fact only 0.85 of its PageRank score. The other 0.15 is split evenly amongst all other nodes. There are now two types of links in the graph, strong and weak. Strong links are the links derived from the dataset. Weak links are artificially created by the damping factor and link all nodes to all other nodes (and itself). This is an important aspect in the calculations which are described herein as it helps convergence of the iterative calculations to a fixpoint.

We avoid needless repetitive calculations by pre-calculating the minimum rank value, which is the value equally distributed to nodes from universal links. Universal links are those resulting from the dampening factor d of the PageRank algorithm. They are $1-d$ of every outlinking (live) nodes score from the previous iteration. Universal links also include even distribution of authority from non-linking (dead) nodes. In some versions of the algorithm these are removed from the graph, but for brevity we leave them in. It is quite common for such nodes to appear in quantity in smaller topical subgraphs.

To summarise the algorithm and give a higher level synopsis (using terms outlined in Table 1), the following steps are conducted in the ranking vector calculation process:

Table 1. Notation used

Constant	**Description**
d	Dampening factor of PageRank calculation: $d = 0.85$
Variable	**Description**
G	Graph to be analysed, represented by a connectivity matrix
λ_1	First eigenvector of G
n	Number of nodes in G
m	Number of links in G
R_k	Ranking score vector for iteration k, approximation of λ_1
i_j	Number of inlinks to node j
o_j	Number of outlinks from node j
in_j	Set of nodes linking node j
out_j	Set of nodes node j links
$dead_G$	Set of dead nodes in G, nodes with no outlinks
$live_G$	Set of live nodes in G, nodes with outlinks
$W(i,j)$	The weightings entryfor link from node i to node j
min_k	The minimum or base rank of a node in iteration k due to universal links

1. **Initialise.** Create an initial eigenvector estimation R_0 with each node i initialised to i_i/m.
2. **Before first iteration and during iterations** The summation of the ranking value of all the dead-link nodes, $\sum_{j \in dead_G} R_k(j)$, and separately, the summation of all the outlinking nodes, $\sum_{j \in live_G} R_k(j)$, are calculated before the first, and during each iteration k. These values are then combined in the following way before iteration k to create a base rank value for each node, min_k.
$$min_k = \frac{\sum_{j \in dead_G} R_{k-1}(j)}{n} + \sum_{j \in live_G} R_{k-1}(j) * \frac{1-d}{n}$$
Also, the old eigenvector is saved for the next iteration's calculations.
3. **During iterations** For each node i, its rank is calculated as follows:
$$R_k(i) = \sum_{j \in in_i} \left(\frac{d}{o_j} * R_{k-1}(j)\right) + min_k$$
If weightings are employed, for each node i, its rank is calculated as follows:
$$R_k(i) = \sum_{j \in in_i} \left(\frac{d}{\sum_{m \in out_j} W(j,m)} * W(j,i) * R_{k-1}(j)\right) + min_k$$
4. **Between every fifth iteration** The algorithm uses quadratic extrapolation [6] between every fifth iteration to speed up convergence.

5. **End of iterations** After each iteration, the l1 norm of the residual is calculated, which is the summation of the absolute change of each value over the iteration. When the L1 residual falls below a certain tolerance, specified as 0.001, the calculations end, and it is considered that a satisfactory estimation R of the dominant eigenvector λ_1 has been found.

The algorithm presented in this section can be used to rank resources in an RDF graph. Consider RDF resources (subject and object in triple notation) as nodes, and RDF properties (predicates) as edges. By applying the presented algorithm and thus examining the link structure of an RDF graph, we achieve rankings for the resources. Please note that only the relevant resources (those which appear as subject at least once in the data set) enter the ranking system, which effectively reduces memory overhead and calculation overhead, and allows output of concise, relevant ranking information. We call this approach ResourceRank. We extend this model by introducing ContextRank and ultimately ReConRank in the next sections. Each use essentially the same algorithm but apply the analysis to different graphs.

4 Extending ResourceRank with Context

In the following, we first show how to apply the ranking algorithm to a context graph, which we call ContextRank. We further describe how we reconcile ResourceRank and ContextRank to ReConRank, a ranking approach which takes into account both the RDF graph itself and the context graph. Context is quite an ambiguous term so to avoid further ambiguity, we define context as the provenance or source of data. This is in-line with the role of context with regards the concept of quads as defined in [4]. Figure 2 shows an example RDF graph of resources, their edges and contexts, outlining an example topical subgraph for the keyword search ‘ReConRank’ with $n = 1$.

Multiple sources can contribute data about the same consolidated resource (by using the same URI). Essentially, this means that anybody can publish data on the web about any resource they like. Whilst this facilitates a strong community driven knowledge base, data from certain contexts may be unreliable or unsuitable for describing a resource. Thus ranking metrics which quantify trust and importance of contexts by exploiting link structures are essential to ranking RDF data. Such metrics can be used for negotiating data from different sources: by the UI component, for example. In this section, we provide a methodology for providing such ranking scores. For now, we must examine in detail the nature of RDF graphs including contexts and resources, and taking an example.

Again, multiple sources or contexts can contribute data about the same resource. This is an interesting aspect of the RDF graph whereby contexts, resources and linkages are represented. When one considers that contexts may also be resources (whereby metadata is provided about the context), and that resources link contexts the exact nature of a graph of contexts and resources can become complex.

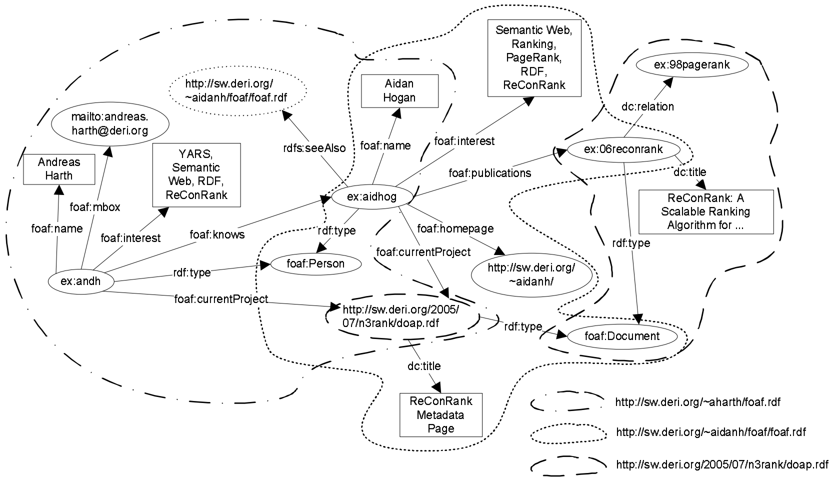
Semantic Web, Ranking, PageRank, RDF, ReConRank
http://sw.deri.org/ ~aidanh/foaf/foaf.rdf
Aidan Hogan
ex:98pagerank
mailto:andreas. harth@deri.org
dc:relation
rdfs:seeAlso
foaf:name
foaf:interest
Andreas Harth
YARS, Semantic Web, RDF, ReConRank
ex:06reconrank
foaf:publications
dc:title
foaf:mbox
ex:aidhog
ReConRank: A Scalable Ranking Algorithm for ...
foaf:name
foaf:interest
foaf:knows
rdf:type
foaf:homepage
rdf:type
foaf:currentProject
foaf:Person
http://sw.deri.org/ ~aidanh/
ex:andh
rdf:type
foaf:currentProject
http://sw.deri.org/2005/ 07/n3rank/doap.rdf
rdf:type
foaf:Document
dc:title
ReConRank Metadata Page
http://sw.deri.org/~aharth/foaf.rdf
http://sw.deri.org/~aidanh/foaf/foaf.rdf
http://sw.deri.org/2005/07/n3rank/doap.rdf

Fig. 2. Three graphs with overlapping context. Resources may also be contexts (which is the case for `http://sw.deri.org/~aidanh/foaf/foaf.rdf` and `http://sw.deri.org/2005/07/n3rank/doap.rdf`).

To be able to perform ResourceRank on the sample graph, we first extract all resources that occur at least once on the subject position of a triple (in fact quad including context), and then calculate rankings with the general ranking algorithm. Figure 3 shows the graph extracted from the graph in Fig. 2 using the ResourceRank approach. Please observe that the extracted graph is not well interlinked.

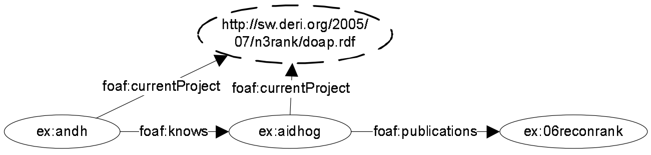

Fig. 3. Link structure of graph without involving contexts. The nodes are not well linked.

To be able to rank the contexts inherent in the RDF graph, we extract the context graph and use that as input to the ranking algorithm. Figure 4 shows the graph extracted from Fig. 2 with only contexts and their linkage. The context graph is not well interlinked. The resulting ContextRank be used to derive prioritisation for references to the source of data and serialisation of data from different sources. However, we now present a means of bringing these

ResourceRank and ContextRank together for the mutual benefit of the results of both. We show how a well-linked graph can be derived by implying links in a certain fashion between contexts and resources, and how analysis of such a graph can benefit the quality of results. We call the derivation of this unified graph and its ranking 'ReConRank'; a unified methodology for RDF entity prioritisation.

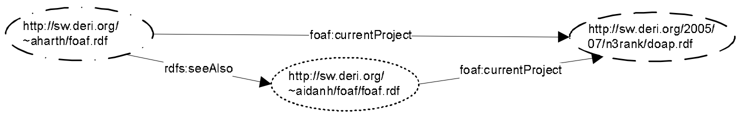

Fig. 4. Link structure of graph without involving resources. Again, the nodes are not well linked.

The graph in Fig. 2 illustrates the explicit relationships inherent in RDF data between contexts and resources. However, relationships can also be implied thus:

1. **Link between context and resource(s) it contains** An implicit link exists between a context and the resource(s) described in that context. PageRank should be propagated from the context (e.g. `http://sw.deri.org/~aidanh/foaf/foaf.rdf`) to its resources (e.g. `ex:aidhog`, `ex:06reconrank`).
2. **Link between resources and containing context(s)** An implicit link also exists between a resource and the context(s) in which it is described. PageRank should flow from the resources (e.g. `ex:aidhog`, `ex:06reconrank`) to their context (e.g. `http://sw.deri.org/~aidanh/foaf/foaf.rdf`).
3. **Link from a context to context** The final type of link implied is whereby a resource in context A links to context B. A link from context A to context B is implied.

Intuitively, for implied links of type 1 and 2, authority should flow between resources and contexts in both directions as

- The importance of contexts relies on the data or resources described within.
- Conversely, resources that appear in important contexts should also be more highly regarded.
- Resources that occur in multiple contexts should also be highly ranked.

The third type of implied link would actually have already been included under links of type 1. However, they are a distinct case and so are regarded separately. The intuition behind them is as follows, a link from a resource in a context can equally be regarded as originating from the context as well as the resource. Figure 5 illustrates the resulting graph, which includes the implied links and isolates only the 'rankable entities' or the contexts and resources which appear as subjects.

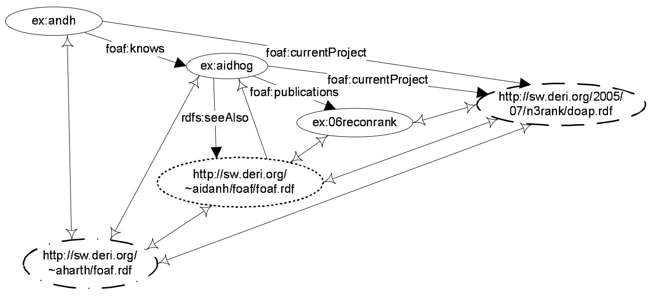

Fig. 5. Link structure of graph for ReConRank analysis after implied links (hollow) are added and rankable resources isolated, as derived from Fig. 2. The nodes are well inter-linked and so amenable to links analysis.

Referring back to the weightings functionality, implied links could be weighted differently according to their type and nature. For instance, an implied link from resource A to context B could be weighted according to the ratio of triples about resource A found in context B. That is, links from resources to contexts would be proportional to the amount of data on the resources provided by each context. Similarly, weightings could be assigned to links from contexts to resources to reflect the portion of data dedicated to a resource by a context. This is yet to be implemented, pending quality evaluation.

Table 2. Results from ranking the the graph represented in Fig. 5.

Context	**ReConRank Score**
http://sw.deri.org/2005/07/n3rank/doap.rdf	0.22870030410893372
http://sw.deri.org/˜aidanh/foaf/foaf.rdf	0.227129832473108
http://sw.deri.org/˜aharth/foaf.rdf	0.18385903708487736
Resource	**ReConRank Score**
http://sw.deri.org/2005/07/n3rank/doap.rdf	0.22870030410893372
ex:06reconrank	0.16572774715471314
ex:aidhog	0.12741015855786822
ex:andh	0.06411971604393776

From applying the algorithm described in Sect. 3 to the graph in Fig. 5, we arrive at two tables of ranks, one of ReConRank metrics for the resources and one of ReConRank metrics for the contexts. Note that there may be overlap between the two whereby a context is also a resource, and will appear in both tables. See Table 2 for results from applying ReConRank to the graph of Fig. 5 (derived from Fig. 2). ResourceRank metrics can then be passed on to a UI component for serialising a results page displaying ranked resources and their associated data.

Table 2 provides scores and priorities for the topical subgraph isolated for the keyword query 'ReConRank'. The top ranked context relating to the topic 'ReConRank' is deemed as `http://sw.deri.org/2005/07/n3rank/doap.rdf`, which is representative in the example of the homepage of the project. Thus, this would indeed be a valuable and trustworthy source of data about the topic, and benefits from containing information on a paper in the area and links from an author.

The second context is `http://sw.deri.org/~aidanh/foaf/foaf.rdf`, the FOAF page of an author in the area. The context is ranked reasonably highly as it provides information related to and/or links to the project homepage, to a paper on the topic and to an author in the area. As the second most valuable source of information in the example for ReConRank resources, its rank corresponds with what would be expected.

Finally, the lowest ranked context is that of `http://sw.deri.org/~aharth/foaf.rdf`, another FOAF page of an author in the area. This is ranked as the least significant source of information on the topic 'ReConRank' as it features only information about two authors in the area, and not any other 'topical nodes'.

Aside from examining the context which also appears as a resource in the ReConRank results (and deservedly so), the top ranked resource is that of a paper in the area: intuitively relevant. Following the paper are the two authors. Note that neither the context `http://sw.deri.org/~aharth/foaf.rdf` nor `http://sw.deri.org/~aidanh/foaf.rdf` appear in the resource rank table. This is because neither are a 'rankable resource' in that in the dataset examined, these contexts do not appear as the subject of a triple. Thus, a results page displaying resources and their corresponding data would not require a rank for this context in relation to ordering these displayed resources. It would solely require the context's rank for prioritising data from that source.

Such a results set would help towards a meaningful serialisation of the data for display to a user. The top presented resources would be the project homepage of the topic queried and the paper on the topic and so on. Information about these resources could then be prioritised by the importance or trustworthiness of the contexts, as numerated by the ranking scores. For example, the UI could use context ranks to serialise individual statements about a resource using varying colours; for instance to display data from poorly linked contexts in light grey and data from multiple well-linked contexts in black. This would offer discerning users a valuable insight into the trustworthiness of the data and matches requirements laid out in Sect. 2.

To re-iterate again, the real power of ReConRank is the representation in the derived graphs of the symbiotic relationship between context authority and resource authority. Applying links based analysis to this unified graph intuitively derives better results than from ranking resource and context graphs individually. Although two different ranking tables are derived, they are calculated from the one graph and so are heavily dependant on one another. The unified ReConRank approach also help solve the issue of analysing poorly linked data. A quick comparison of Fig. 5 to Fig. 3 and Fig. 4 illustrates the considerable improvement in the quality of the graph for link analysis with the ReConRank methodology.

5 Experimental Evaluation

The above algorithms are implemented in Java and accept Notation3[1] as input data format (with additional entry for context). Some source code is available online for download[2].

We conducted the experiments on a machine with a single Opteron 2.2 GHz CPU, 4 GB of main memory and two SATA 160GB disks running Debian Linux with Sun's JVM in version 1.5.0_06-b05. Maximum JVM heap size was set to 2GB which proved sufficient for analysing the full dataset.

The dataset was obtained from the Web with a crawler following `a href` links in HTML and `rdfs:seeAlso` links in RDF. HTML and XML documents were transformed to RDF before inclusion in the dataset. The entire dataset consists of around 70k files which results in a total of around 15M quads. The file size of the combined dataset is 2.8GBytes and contains 2.618 million unique (rankable) resources and contexts.

Figure 6 shows the performance results for partitions of the dataset in varying sizes. The average number of iterations was 29.6, with a tolerance for the l1 norm of the residual set at 0.001.

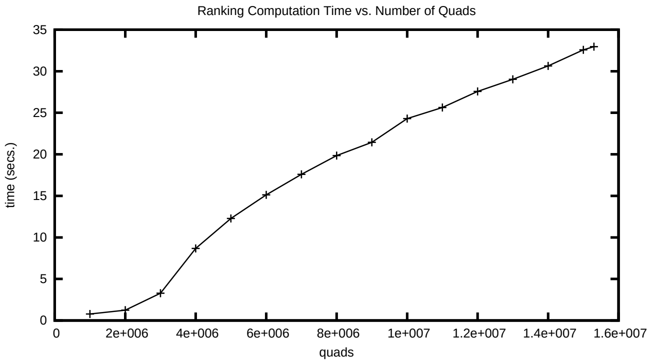

Fig. 6. Performance of the ReConRank algorithm applied to a 15M triple dataset obtained from the Web. We measured the performance of the algorithm for parts of the dataset with varying size.

[1] `http://www.w3.org/DesignIssues/Notation3`

[2] `http://sw.deri.org/2005/07/n3rank/`

6 Related Work

Google, as is widely known, have enjoyed much success in the area of web searching through use of PageRank [8], a links based analysis technique and other ranking techniques. PageRank is applied to the web graph of webpages hyperlinking to each other. It is a proven scalable algorithm and a mature research area. Our approach illustrates a means of adapting PageRank for use in RDF data.

Kleinberg [7] uses the notion of focussed subgraphs which we extend to topical subgraphs on RDF data.

ObjectRank [5] describes an approach to rank a directed labelled 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. No consideration is given to provenance of data.

Damian et al. [2] assign trust and ranking metrics to peers within a social network and to desktop resources of the peers. Evaluation was applied to a use-case of some instances of CiteSeer data distributed on small numbers of peers. They deal with context information, however they loosely define context as specific aspects of a situation and not as provenance of data.

Halaschek et al. [3] use the term context but refer to topics of RDF graphs, not the provenance of information as in our model. Their system can find relationships between two entities in a database. Some consideration is given to trust metrics regarding data from different sources, however the system depends on user input to specify the trustworthiness of a source.

Bamba et al. [1] have also documented work in a similar area to this paper, however they employ a HITS type architecture computing two metrics to correspond to hubs and authorities which they call subjectivity and objectivity respectively. They apply such analyses to what they call an 'isAGraph' of `rdf:Class` nodes linked by `rdfs:subClassOf`. They also analyse 'propertyGraphs', derived from resources linking through being the subject and object of a triple. Thus, they rank both classes and resources, but do not factor in provenance linkage.

7 Conclusion

We have shown how to adapt PageRank to Semantic Web data while taking into account context. The system performs well for large datasets, although we expect the typical size of the graph to be ranked relatively small due to the topical subgraph approach. Thus we conclude that ReConRank as outlined herein can be integrated into a scalable system, with minimal effect on performance or scalability, to provide meaningful rank metrics to result sets which can be computed at query-time.

We have yet to conduct an extensive evaluation of the quality of the ranking results provided under this methodology. The methodology is intended for use within the Semantic Web Search Engine architecture, which upon being developed should allow users to evaluate the results ordering. For now, one must be sated that the intuition outlined behind the methodology holds, and that the illustrated example of the analysis applied to the 'ReConRank' topical subgraph reveals quite accurate results.

Finally, we end by concluding that the methodology outlined herein is a step towards a unified RDF ranking scheme, whereby contexts are ranked alongside resources. Thus results from the approach offer not only metrics on the importance of resources within a relevant topic; but also a quantification of the trustworthiness or importance of a context as a source of data on that topic. By combining the two under ReConRank, the results of both are enhanced.

References

1. B. Bamba and S. Mukherjea. Utilizing resource importance for ranking semantic web query results. In *Semantic Web and Databases, Second International Workshop, SWDB 2004, Toronto, Canada, August 29-30, 2004, Revised Selected Papers*, pages 185–198, 2004.
2. A. Damian, W. Nejdl, and R. Paiu. Peer-sensitive objectrank - valuing contextual information in social networks. In *Web Information Systems Engineering - WISE 2005, 6th International Conference on Web Information Systems Engineering, New York, NY, USA, November 20-22, 2005, Proceedings*, pages 512–519, 2005.
3. C. Halaschek, B. Aleman-Meza, I. Arpinar, and A. Sheth. Discovering and ranking semantic associations over a large rdf metabase, 2004.
4. A. Harth and S. Decker. Optimized index structures for querying rdf from the web. In *Proceedings of the 3rd Latin American Web Congress*, pages 71–80. IEEE Press, 2005.
5. H. Hwang, V. Hristidis, and Y. Papakonstantinou. Objectrank: a system for authority-based search on databases. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 796–798, New York, NY, USA, 2006. ACM Press.
6. S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Extrapolation methods for accelerating pagerank computations. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 261–270, New York, NY, USA, 2003. ACM Press.
7. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
8. 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.
9. M. Richardson and P. Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.
10. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.