A method to rank nodes in an RDF graph

Alvaro Graves
Dept. of Computer Science
Rensselaer Polytechnic Inst.
agraves@cs.rpi.edu

Sibel Adalı
Dept. of Computer Science
Rensselaer Polytechnic Inst.
sibel@cs.rpi.edu

James Hendler
Dept. of Computer Science
Rensselaer Polytechnic Inst.
hendler@cs.rpi.edu

ABSTRACT

Despite the increasing popularity of RDF as a data representation method, there is no accepted measure of the importance of nodes in an RDF graph. Such a measure could be used to sort the nodes returned by a SPARQL query or to find the important concepts in an RDF graph. In this paper we propose a graph-theoretic measure called *noc-order* for ranking nodes in RDF graphs based on the notion of centrality. We illustrate that this method is able to capture interesting global properties of the underlying RDF graph using study cases from different knowledge domains. We also show how well noc-order behaves even if the underlying data has some noise, i.e. superfluous and/or erroneous data. Finally, we discuss how information about the importance of different predicates either based on their informativeness, prior semantic information about them or user preferences can be incorporated into this measure. We show the effects of such modifications to the ranking method by examples.

1. INTRODUCTION

Resource Description Framework (RDF) has recently emerged as a popular data model in many different applications, from scientific and business domains to social networking applications. The increasing acceptance of more expressive ontology languages such as the various versions of OWL is expected to continue this trend and lead to development of large data sets with complex graph structures. The effective use of these data sets requires new tools for organizing the content. Currently, nodes are displayed without any kind of sort which can lead very long unordered list of nodes in large RDF datasets. Such a method can also be used to find the important concepts in an RDF data set or discover problems in the data set (for example for testing whether prominent concepts are well-represented or not). We developed a method called *noc-order* (pronounced as knock-order, based on *NOde Centrality Ordering*) to rank nodes in an RDF data set by treating the information in the data set as a labeled undirected graph.

We base our ranking in the concept of "closeness centrality" which measures how costly it is to reach all the nodes in the graph from one specific node. As a motivation, we use the analogy of a traveler who does not know where to go and has a limited budget (time or money). Suppose also that she can start her trip from many different places on the map. Where should she start? One possible criteria is to choose a central location from where it is possible to reach to many other (interesting) places with a low cost. This way, not knowing where she should go next, she has the highest number of options. Furthermore, the closer the attractions are, the better it would be. This way, there are more options at each leg of the trip as well. We consider exploring nodes in an RDF graph in a similar manner. If a query is expected to return nodes, it would be better to return nodes that reach many other nodes in as few hops as possible. This way, the explorer examining the results can explore larger portions of the graph quickly. To model this measure, we use the notion of centrality of a node as the ranking criteria. The more central the node is, easier it is to reach the rest of the graph. We also show that we can incorporate other factors to this rank, such as "interestingness" of the roads or the desirability of the short/long paths, etc.

2. CALCULATING *NOC-ORDER*

The measure of centrality is equivalent to computing the *All-Pairs Shortest Path* problem. This problem can be solved using different approaches [4][7][9][10] and can also be approximated efficiently [5]. In order to calculate the *noc-order* ranking we need to specify certain graph properties and to define a measure of distance.

In order to calculate the centrality of each node, we found that it is useful to consider the RDF graph as an undirected, labeled graph. This is mainly because we are more interested in how the nodes are connected rather than the direction of the semantics. Also, it is always possible to find a semantic inverse of a predicate (e.g. `fatherOf` as opposite of `sonOf`).

Definition of distance: We use three main definitions of distance. First we define $d_0(x, y, p) = m$ where m is the number of edges in path p (this is the usual distance function). Also we use $d_w(x, y, p) = \sum_{i=1}^{m} w(e)$ is the sum of the weights of the edges in the path based on a given weight function w such that $w : E \rightarrow \Re$. This weight can based on many different factors or learnt from user feedback. In this work, we consider the frequency of occurrence of each predicate for weights: thus, the more common a predicate is, the less relevant for our ranking. Finally we define $d_w^{\alpha}(x, y, p) = \sum_{i=1}^{m}(w(e)/\alpha^i)$ where the distance gets longer as a function of α where $0 \leq \alpha \leq 1$, i.e. the shorter paths are even more desirable over longer ones. We then normalize the distance by the number of nodes reachable by the given node. Note that this order favors more central nodes. However, it is as easy to create the reverse ordering to favor the edge cases.

Connectedness: We have found that in several cases, the RDF graph could be disconnected. Specifically, it is possible to find several examples that consists of a big component and one or more smaller components. These components are at least one or two order of magnitudes smaller than the main component. In these cases, we have to further penalize the nodes that belong to the smaller components, otherwise the nodes in the smaller components would be more central, since the cost to reach all the nodes in their components would be —in general— smaller. Another approach we use is to take only the main connected component as the subject of study, since that in most of the cases it contains more than 99% of the nodes.

3. DATASETS

We have tested the *noc-order* ranking against several different datasets that belongs to very different domains.

- **CIA Factbook:** This dataset contains information and facts about different countries of the world. It contains more than 30,000 nodes and 98,000 edges.
- **Terrorist Ontology:** Represents information about people, countries and events related to terrorism, among others. It contains more than 14,000 nodes and 33,000 edges.
- **Wine Ontology:** Contains information about different types of wines, characteristics, regions, among others. It contains 720 nodes and almost 2,000 edges.

Table 1: Comparison between (a) the top 10 and (b) bottom 10 nodes in CIA Factbook using in-degree and centrality ranking.

(a)

In-degree ranking	Centrality ranking
rdf:Statement	org:IOC
cia:Estimate	org:WHO
cia:CommodityPercent	org:UN
cia:CountryPercent	org:ITU
cia:AirportBreakdown	org:UNCTAD
cia:Port	org:UPU
cia:SexRatioBreakdown	org:ICAO
cia:EthnicGroupPercent	org:UNESCO
cia:ReligionPercent	env:Biodiversity
cia:LanguagePercent	env:Climate_Change

(b)

In-degree ranking	Centrality ranking
cia:A110217	work:priests
cia:A110234	work:nuns
cia:A110239	work:guards
cia:A110323	work:3,000_lay_workers_live _outside_the_Vatican
cia:A110329	work:animal_husbandry_and _subsistence_farming
cia:A110335	work:wholesale_and_retail_distr.
cia:A110341	work:prof._and_scientific_serv.
cia:A110360	work:misc._services
cia:A110365	work:forestry_and_fishing
cia:A110370	work:entertainment_and_catering

4. RESULTS

In our implementation, we translate the datasets into the N-Triples [6] format using Raptor RDF parser library [8]. First we find the connected components, then we run Dijsktra's algorithm [3] to find the shortest path for each node in the graph using one of the distance functions described earlier. Using this strategy allowed us to parallelize the problem, improving the speed and to interrupt the whole computation to study the partial results and being able to resume it at will. As an example, the case of the CIA Factbook dataset we show the results obtained with our ranking and compared with a ranking made using the in-degree distribution of the predicates. The results can be seen in table 1. It is easy to see that for the in-degree ranking, the best ranked nodes belongs usually to the schema. In our ranking however we find nodes representing international organizations and some general-interest topics, like "climate change". Hence, the *noc-order* is able to find the central concepts in that specific data set. A similar trend can be seen in all our tests, the higher rank nodes are more general and central concepts while the lower ranked nodes are the most specific concepts, allowing the system to tailor the ranking accordingly.

5. CONCLUSIONS AND FUTURE WORK

In this work, we have introduced a simple ranking method for nodes called the *noc-order* based on centrality in an RDF graph. Our ranking method can be used to rank RDF nodes or combined with other methods to answer different types of queries [1][2]. In future work, we would like to run our ranking in big FOAF datasets for validation. Also, it would be interesting to study the effect of different weight functions and α values in the ranking. We are also investigating how to incorporate the semantics in OWL ontologies to improve the ranking and incorporate user feedback into the ranking method.

6. REFERENCES

[1] B. Aleman-Meza. Searching and ranking documents based on semantic relationships. *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, page 5, Mar 2006.

[2] B. Aleman-Meza, C. Halaschek-Weiner, and I. Arpinar. Ranking complex relationships on the semantic web. *IEEE Internet Computing*, Jan 2005.

[3] T. H. Cormen, C. E. Leiserson, , R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

[4] E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, Jan 1959.

[5] D. Eppstein and J. Wang. Fast approximation of centrality. *Proceedings of the twelfth annual ACM-SIAM symposium on ...*, Jan 2001.

[6] N.-T. format. www.w3.org/tr/rdf-testcases/#ntriples.

[7] D. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM (JACM)*, 24(1):1–13, 1977.

[8] R. R. parser library. librdf.org/raptor/.

[9] R. Seidel. On the all-pairs-shortest-path problem in unweighted undirected graphs. *Journal of Computer and System Sciences*, page 4, Oct 1999.

[10] J. Sibeyn. External matrix multiplication and all-pairs shortest path. *Information Processing Letters*, Jan 2004.