A Performance and Scalability Metric for Virtual RDF Graphs

Michael Hausenblas[1], Wolfgang Slany[2], and Danny Ayers[3]

[1] Institute of Information Systems and Information Management,
JOANNEUM RESEARCH, Steyrergasse 17, 8010 Graz, Austria
michael.hausenblas@joanneum.at
[2] Institute for Software Technology,
Graz University of Technology, Inffeldgasse 16b, 8010 Graz, Austria
wsi@ist.tugraz.at
[3] Independent Author, Italy
danny.ayers@gmail.com

Abstract. From a theoretical point of view, the Semantic Web is understood in terms of a stack with RDF being one of its layers. A Semantic Web application operates on the common data model expressed in RDF. Reality is a bit different, though. As legacy data has to be processed in order to realise the Semantic Web, a number of questions arise when one is after processing RDF graphs on the Semantic Web. This work addresses performance and scalability issues (PSI), viz. proposing a metric for *virtual RDF graphs on the Semantic Web*—in contrast to a local RDF repository, or distributed, but native RDF stores.

1 Motivation

The Semantic Web is—slowly—starting to take-off; as it seems, this is mainly due to a certain pressure stemming from the Web 2.0 success stories. From a theoretical point of view the Semantic Web is understood in terms of a stack. The Resource Description Framework (RDF) [1] is one of the layers in this stack, representing the common data model of the Semantic Web.

However, practice teaches that this is not the case, in general. In the perception of the Semantic Web there exists a tremendous amount of legacy data. This is, HTML pages with or without microformats[4] in it, relational databases (RDBMS), various XML applications as Scalable Vector Graphics (SVG)[5], and the like. These formats are now being absorbed into the Semantic Web by approaches as Gleaning Resource Descriptions from Dialects of Languages (GRDDL) [2], RDFa in HTML [3], etc.

Take Fig. 1 as an example for a real-world setup of a Semantic Web application. There, a Semantic Web agent operates on an RDF graph with triples that actually originate from a number of sources, non-RDF or 'native' RDF alike.

[4] http://microformats.org/
[5] http://www.w3.org/Graphics/SVG/

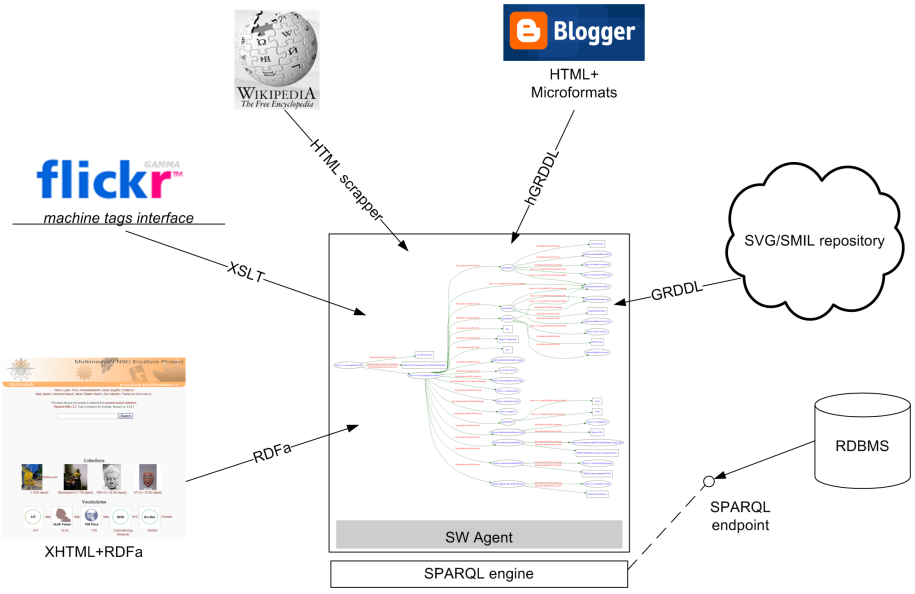


Fig. 1. A real-world setup for a Semantic Web application.

The Semantic Web agent operates on its local RDF graph in terms of performing for example a SPARQL Protocol And RDF Query Language [4] query to accomplish a certain task. While from the point of view of the SPARQL engine it might not be of interest where triples come from and how they found their way into the local RDF graph, the Semantic Web agent—and finally the human user who has instructed the agent to carry out a task—may well be interested in how long a certain operation takes.

But how do the triples arrive in the local RDF graph? Further, how do the following issues influence the performance and the scalability of the operations on the local RDF graph:

- The number of sources that are in use;
- The types of sources, as RDF/XML, RDF in HTML, RDBMS, etc.;
- Characteristics of the sources: a fixed number of triples vs. dynamic, as potentially in case of a SPARQL end point.

This paper attempts to answers these questions. We first give a short overview of related and existing work, then we discuss and define virtual RDF graphs, their types, and characteristics. How to RDF-ize the flickr Web API[6], based on the recently introduced *machine tags*[7] feature, serves as a showcase for the proposed metric. Finally we conclude on the current work and sketch directions for further investigations.

[6] `http://www.flickr.com/services/api/`
[7] `http://www.flickr.com/groups/api/discuss/72157594497877875/`

2 Related and Existing Work

The term scalability is used differently in diverse domains—a generic definition is not available [5]. For selected domains, various views on scalability are available [6, 7]. Bondi [8] recently elaborated on scalability issues. He considers four types of scalability: *load scalability*, *space scalability*, *space-time scalability*, and *structural scalability*.

Practice-oriented research regarding RDF stores has been reported in the SIMILE[8] project [9], and in a joint W3C-EU project, Semantic Web Advanced Development for Europe [10]. In the Advanced Knowledge Technologies (AKT) project, 3store [11]—a scalable RDF store based on Redland[9]—was used. A framework for testing graph search algorithms where there is a need for storage that can execute fast graph search algorithms on big RDF data is described in [12].

At W3C, RDF scalability and performance is an issue[10], though the scope is often limited to local RDF stores. There are also academic events that address the scalability issue, such as the International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS)[11] which took place the second time in 2006.

Some research already exists regarding distributed RDF stores. Though the focus is on distributed RDF repositories, it is always assumed that one is dealing with 'native' RDF sources. An excellent example is the work of Stuckenschmidt et. al. [13]. They present an architecture for optimizing querying in distributed RDF repositories by extending an existing RDF store (Sesame). Cai and Frank [14] report on a scalable distributed RDF repository based on a peer-to-peer network—they propose RDFPeers that stores each triple at three places in a multi-attribute addressable network by applying globally known hash functions. From the Gnowsis project, a work has been reported that comes closest to ours: Sauermann and Schwarz [15] propose an adapter framework that allows for integrating data sources as PDFs, RDBMS, and even Microsoft Outlook.

To the best of our knowledge no research exists that addresses the issue of this paper, i.e. performance and scalability issues of virtual RDF graphs on the Semantic Web. Regarding the distributed aspect, the authors of [13] listed two strong arguments, namely i) the *freshness*, viz. in not using a local copy of a remote source frees one from the need of managing changes, and ii) the gained *flexibility*—keeping different sources separate from each other provides a greater flexibility concerning the addition and removal of sources. We subscribe to this view and add that there are a number of real-world use cases which can only be addressed properly when taking distributed sources into account. These use cases are to be found in the news domain, stock exchange information, etc.

---

[8] `http://simile.mit.edu`

[9] `http://librdf.org/`

[10] `http://esw.w3.org/topic/TripleStoreScalability`, and `http://esw.w3.org/topic/LargeTripleStores`

[11] `http://www.cs.vu.nl/~holger/ssws2006/`

3 Virtual RDF Graphs

To establish a common understanding of the terms used in this paper, we first give some basic definitions. We describe what a Semantic Web application in our understanding is, and have a closer look at virtual RDF graphs. The nature of virtual RDF graphs, and their intrinsic properties are discussed in detail.

Definition 1 (Semantic Web application). *A **Semantic Web application** is a software program that meets the following minimal requirements:*

1. *It is based on, i.e. utilises HTTP*[12] *and URIs*[13]*;*
2. *For human agents, the primary presentation format is (X)HTML*[14]*, for software agents, the primary interface formats are based on Web services*[15] *and/or based on the REST approach [16];*
3. *The application operates on the Internet; the number of concurrent users is undetermined.*
4. *The content used is machine readable and interpretable; the data model of the application is RDF [1].*
5. *A set of formal vocabularies—potentially based on OWL [17]—is used to capture the domain of discourse. At least one of the utilised vocabularies has to be proven **not** to be under full control of the Semantic Web application developer.*
6. ***Non-mandatory**, SPARQL [4] is in use for querying, and RIF [18] for representing, respectively exchanging rules.*

The restriction that a Semantic Web application is expected to operate on the Internet is to ensure that Intranet applications that utilise Web technologies are **not** understood as Semantic Web applications in the narrower sense. This is a matter of who controls the data and the schemes rather than a question of the sheer size of the application.

Definition 1 requires that a Semantic Web application operates on the RDF data model, which leads us to the virtual RDF graph, defined as follows.

Definition 2 (Virtual RDF Graph). *A **virtual RDF graph** (vRDF graph) is an RDF graph local to a Semantic Web application that contains triples from potentially differing, non-local sources. The primary function of the vRDF graph is that of enabling CRUD*[16] *operations on top of it. The following is trivially true for a vRDF graph:*

1. *it comprises actual source RDF graphs (henceforth sources), with N_{src} being the number of sources;*
2. *each source S^i_{src} contributes a number of triples T^i_{src} to a vRDF graph, with $0 < i \leq N_{src}$;*
3. *The vRDF graph contains $\sum T^i_{src}$ triples.*

[12] `http://www.w3.org/Protocols/rfc2616/rfc2616.html`
[13] `http://www.ietf.org/rfc/rfc2396.txt`
[14] `http://www.w3.org/html/wg/`
[15] `http://www.w3.org/2002/ws/`
[16] **c**reate, **r**ead, **u**pdate and **d**elete—the four basic functions of persistent storage

3.1 Types Of Sources

Triples may stem from sources that utilise various representations. In Fig. 2 the representational properties of the sources are depicted, ranging from the RDF model[17] to non model-compliant sources.

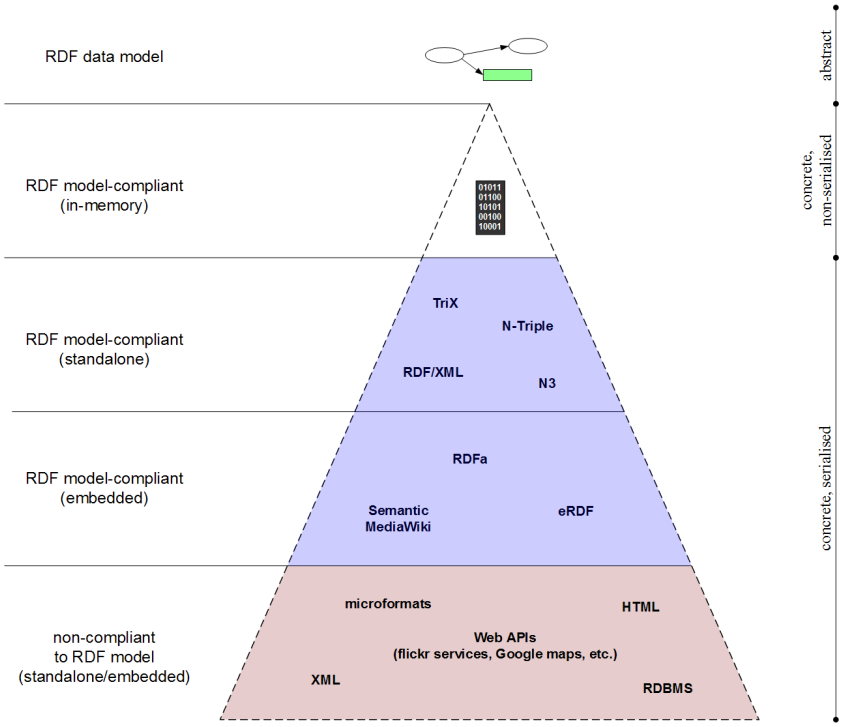


Fig. 2. The RDF representation pyramid.

The two middle layers of the pyramid denote representations that are RDF model-compliant and have a serialisation, hence may be called *native RDF*. Representations that do have a serialisation, but are not RDF model-compliant, may be referred to as *non-RDF sources*. We therefore differentiate:

Standalone, RDF model-compliant Representations. These type of sources, for example RDF/XML, can be stored, transmitted, and processed on their own. For example an in-memory Document Object Model (DOM) representation of an RDF/XML document can be built by utilising a SAX parser.

Embedded, RDF model-compliant Representations. Sources of this type, as RDFa in HTML [3] or eRDF[18], need a host to exist; their representation

[17] `http://www.w3.org/TR/rdf-concepts/#section-data-model`

[18] `http://research.talis.com/2005/erdf/wiki/Main/RdfInHtml`

is only defined in the context of this host. Here, the triples are produced by applying a transformation.

Representations non-compliant to the RDF model. The majority of the data sources on the Web, standalone or embedded, is of this type:

- GRDDL [2] is utilised to 'uncover' RDF in, e.g., HTML. The same applies to microformats that can be RDF-ized using hGRDDL[19];
- An RDBMS that provides for a SPARQL end point [4] can be used to contribute triples[20];
- Syndicated feeds (RSS 2.0, Atom[21]) are another source;
- From a HTML page without explicit metadata, triples may be gathered through screen scrapers (as [19]).

In order to be processed, the serialisation is required to be "converted" from a representation with a concrete syntax into an in-memory representation. This conversion may occur through applying a transformation[22], or by parsing the specified syntax.

3.2 Characteristics Of Sources

Besides the type of the source, a further distinction w.r.t. the number of triples from a source can be made. We distinguish between fixed sized sources and undetermined—or dynamic—sized source.

Take for example a Wiki site that serves as a source for a vRDF graph. Let us assume an HTML scraper is used to generate triples from selected Wiki pages, for example based on a category. The number of resulting triples then is in many cases stable and can be assessed in advanced. In contrast to this, imagine an RDBMS that provides for a SPARQL end point—the DR2 Server[23] is a prominent example for this—as an example for a dynamic source. Based on the query, the number of triples varies.

4 A Metric for virtual RDF Graphs

In this section we describe a performance and scalability metric that helps a Semantic Web application developer to assess her/his vRDF graph. A showcase for a non-native RDF source is then used to illustrate the application of the metric.

The execution time of an operation on a vRDF graph is influenced by a number of factors, including the number of sources in a vRDF graph N_{src}, the overall number of triples $\sum T^i_{src}$, and the type of the operation. The metric proposed in Definition 3 can be used to assess the performance and scalability of an vRDF graph.

[19] `http://www.w3.org/2006/07/SWD/wiki/hGRDDL_Example`

[20] Though, this source may also be considered as being native in terms of the interface.

[21] `http://bblfish.net/work/atom-owl/2006-06-06/AtomOwl.html`

[22] see `http://esw.w3.org/topic/ConverterToRdf`, and also `http://esw.w3.org/topic/CustomRdfDialects`

[23] `http://sites.wiwiss.fu-berlin.de/suhl/bizer/d2r-server/`

Definition 3 (Execution Metric). *The overall execution time for performing a CRUD function (as inserting a triple, performing a SPARQL ASK query, etc.) is denoted as t_P; the time for converting a non-RDF source representation into an RDF graph is referred to as t_{2RDF}. The total time delays due to the network (Internet) transfer are summed up as t_D; the time for the actual operation performed locally is denoted as t_O. Obviously,*

$$t_P = t_O + t_{2RDF} + t_D$$

The "conversion time vs. the overall execution time"-ratio is defined as

$$coR = \frac{t_{2RDF}}{t_P}$$

To illustrate the above introduced metric, a showcase has been set up, which is described in the following.

A Showcase For Non-Native RDF Sources: PSIMeter[24]. The showcase demonstrates the application of the metric by RDF-izing the flickr API. Three different methods have been implemented; the non-native RDF Source used in the PSIMeter showcase is the information present in the machine tags. The goal for each of the three methods is to allow a Semantic Web agent to perform a SPARQL construct statement, as for example:

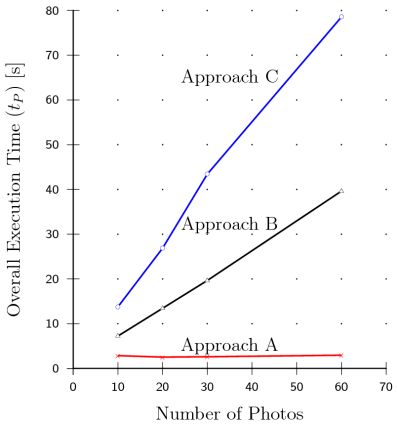
```
CONSTRUCT { ?photoURL dc:subject ?subject }
WHERE     { ?photoURL dc:subject ?subject.
              FILTER regex(?subject, "XXX", "i") }
```

The experiments to compare the three approaches were run on a testbed that comprised up to 100 photos from a single user, along with annotations in the form of machine tags, up to 60 in total. Machine tags were selected as the source due to their straightforward mapping to the RDF model. The three methods for constructing the vRDF graph work as follows:

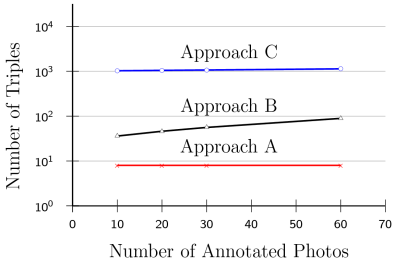
1. *Approach A* uses the search functionality of the flickr API[25] in a first step to retrieve the IDs of photos tagged with certain machine tags. In a second step the flickr API is used to retrieve the available metadata for each photo. Finally the result of the two previous steps is converted into an RDF representation, locally.
2. *Approach B* uses the flickr API to retrieve all public photos[26] firstly. It then uses a local XSL transformation to generate the RDF graph;
3. *Approach C* retrieves all public photos, as in Approach B. Then, for each photo an external service[27] is invoked to generate the RDF graph.

[24] available at `http://sw.joanneum.at:8080/psimeter/`
[25] `http://www.flickr.com/services/api/flickr.photos.search.htm`
[26] `http://www.flickr.com/services/api/flickr.people.getPublicPhotos.html`
[27] `http://www.kanzaki.com/works/2005/imgdsc/flickr2rdf`

Firstly, for a *fixed query*, `dc:subject=marian`, the overall execution time t_P has been measured depending on the number of photos (Fig. 3(a)). In Fig. 3(b), the size of the vRDF graph in relation to the number of annotations, with a fixed number of photos, is depicted.



(a) Overall execution time dependency on number of photos.



(b) vRDF graph size dependency on number of annotated photos.

Fig. 3. PSIMeter: Metric for a fixed query.

The second experiment focused on the impact of the *query type* on the overall execution time.

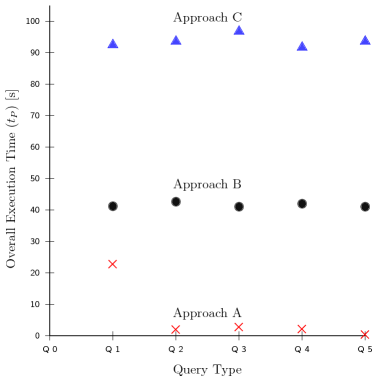


Fig. 4. PSIMeter: Metric in dependency on Query Type.

The results of the second experiment are depicted in Fig. 4, with the according queries listed in Table 1.

Reference	Query
Q1	`dc:subject=dummy`
Q2	`dc:title=NM2`
Q3	`dc:title=`
Q4	`geo:location=athens`
Q5	`dc:dummy=test` (empty match)

Table 1. Query Types

Note, that more than 80% of the photos were tagged with `dc:subject=dummy`, hence Q1 exhibits an exception.

Another finding of the experiment, was that in all evaluation runs, *coR* tended towards 1 (ranging from 0.95 to 0.99), viz. most of the time the system was busy converting the data to RDF, and only a small fraction was dedicated to the actual operation of applying the SPARQL construct statement.

5 Conclusion

When building Semantic Web applications, it is not only important to operate on distributed RDF graphs by means of virtue, but also to question how the triples in a vRDF graph where produced. We have looked at variables that influence the performance and scalability of a Semantic Web application, and proposed a metric for vRDF graphs. As all types of sources must be converted into an in-memory representation in order to be processed, the selection of the type of sources is crucial. The experiments highlight the importance to use existing search infrastructure, as the flickr search API in our case, as far as possible, hence converting only results to RDF.

Another generic hint is to avoid conversion cascades. As long as there exists a direct way to create an in-memory representation, this issue does not play a vital role. Though, regarding performance issues, this is of importance in case an intermediate is used to create the in-memory representation, as with, e.g., the hGRDDL approach.

Finally, the incorporation of dynamic sized sources is a challenge one has to carefully implement. This is a potential area for further research in this field.

6 Acknowledgements

The research reported in this paper was carried out in two projects: the "Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content" (K-Space) project[28], partially funded under the 6th Framework Programme of the European Commission, and the "Understanding Advertising" (UAd) project[29], funded by the Austrian FIT-IT Programme.

[28] `http://kspace.qmul.net/`

[29] `http://www.sembase.at/index.php/UAd`

References

1. G. Klyne, J. J. Carroll, and B. McBride. RDF/XML Syntax Specification (Revised). `http://www.w3.org/TR/rdf-concepts/`, 2004.
2. D. Connolly. Gleaning Resource Descriptions from Dialects of Languages (GRDDL). `http://www.w3.org/TR/2007/WD-grddl-20070302/`, 2007.
3. M. Hausenblas and B. Adida. RDFa in HTML Overview. `http://www.w3.org/2006/07/SWD/RDFa/`, 2007.
4. E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. `http://www.w3.org/TR/rdf-sparql-query/`, 2006.
5. M. D. Hill. What is scalability? *SIGARCH Comput. Archit. News*, 18(4):18–21, 1990.
6. P. Jogalekar and M. Woodside. Evaluating the Scalability of Distributed Systems. *IEEE Trans. Parallel Distrib. Syst.*, 11(6):589–603, 2000.
7. E. L. Miller, D. S., J. Liu, and C. Nicholas. Performance and scalability of a large-scale N-gram based information retrieval system. *Journal of Digital Information (online refereed journal)*, 2000.
8. A. B. Bondi. Characteristics of scalability and their impact on performance. In *WOSP '00: Proceedings of the 2nd International Workshop on Software and Performance*, pages 195–203, New York, NY, USA, 2000. ACM Press.
9. R. Lee. Scalability Report on Triple Store Applications. `http://simile.mit.edu/reports/stores/`, 2004.
10. D. Beckett. Deliverable 10.1: Scalability and Storage: Survey of Free Software/Open Source RDF storage systems. Technical report, Semantic Web Advanced Development for Europe (SWAD-Europe), IST-2001-34732, 2002.
11. S. Harris and N. Gibbins. 3store: Efficient bulk RDF storage. In *Proc. 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), Sanibel Island*, pages 1–15, 2003.
12. M. Janik and K. Kochut. BRAHMS: A WorkBench RDF Store and High Performance Memory System for Semantic Association Discovery. In *Proc. 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland*, pages 431–445, 2005.
13. H. Stuckenschmidt, R. Vdovjak, G.-J. Houben, and J. Broekstra. Index Structures and Algorithms for Querying Distributed RDF Repositories. In *WWW '04: Proc. of the 13th International Conference on World Wide Web*, pages 631–639, New York, NY, USA, 2004. ACM Press.
14. M. Cai and M. Frank. RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network. In *WWW '04: Proc. of the 13th International Conference on World Wide Web*, pages 650–657, New York, NY, USA, 2004. ACM Press.
15. L. Sauermann and S. Schwarz. Gnowsis Adapter Framework: Treating Structured Data Sources as Virtual RDF Graphs. In *Proc. 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland*, 2005.
16. R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
17. OWL. Web Ontology Language Reference. W3C Recommendation, 10 February 2004.
18. Rule Interchange Format (RIF). `http://www.w3.org/2005/rules/`, 2007.
19. R. Baumgartner, G. Gottlob, M. Herzog, and W. Slany. Annotating the Legacy Web with Lixto. In *Proc. 3rd International Semantic Web Conference, ISWC2004, Hiroshima, Japan*, 2004.