FABiIT – finding answers in a billion triples

Conference or Workshop Item

FABilT – Finding Answers in a Billion Triples

Mathieu d'Aquin and Vanessa Lopez and Enrico Motta

Knowledge Media Institute (KMi)
The Open University, Milton Keynes, United Kingdom
{m.daquin,v.lopez,e.motta}@open.ac.uk

Abstract. This submission presents the application of two coupled systems to the Billion Triples Challenge. The first system (Watson) provides the infrastructure which allows the second one (PowerAqua) to pose natural language queries to the billion triple datasets. Watson is a gateway to the Semantic Web: it crawls and indexes semantic data online to provide a variety of access mechanisms for human users and applications. We show here how we indexed most of the datasets provided for the challenge, thus obtaining an infrastructure (comprising web services, API, web interface, etc.) which supports the exploration of these datasets and makes them available to any Watson-based application. PowerAqua is an open domain question answering system which allows users to pose natural language queries to large scale collections of heterogeneous semantic data. In this paper, we discuss the issues we faced in configuring PowerAqua and Watson for the challenge and report on our results. The system composed of Watson and PowerAqua, and applied to the Billion Triples Challenge, is called FABilT.

1 Introduction

The one billion triple datasets form a huge information resource that is not only challenging because of its scale, but also because of its heterogeneity and complexity. In other words, while these datasets literally contain the answers to millions of questions, locating and exploiting the relevant information to extract from them these answers is a major challenge. Here we propose two complementary systems which provide the basic components for realizing this task: finding answers in a billion triple repository.

Watson [1] is a gateway to the Semantic Web: it collects, analyses and indexes online semantic data to provide a variety of mechanisms for users and applications to explore and exploit these data. Watson implements a robust and efficient infrastructure for developing applications able to dynamically make use of the Semantic Web, as a large scale and distributed base of heterogeneous knowledge. For this contest, we created a new instance of Watson indexing most of the datasets provided as part of the "billion triples". The concrete realization of this billion triple Watson covers two aspects:

User interface: We provide the standard Watson interface, allowing users to search and explore the ontologies and semantic documents online. This interface has been designed to facilitate the selection of semantic documents

for reuse, providing advanced navigation features for making sense of the data available.

API: While providing a front-end for human users is important, the core of Watson is a set of web services and the corresponding API that make possible the development of applications exploiting the Watson collection. The billion triple Watson provides all the functionalities required to locate, explore and exploit semantic information contained in the billion triple datasets without the need for an additional, heavy infrastructure to support the application.

One of the strengths of Watson is that it is not designed in a way isolated from other systems, but directly in relation with the applications it intends to support. In this submission, we describe one of these applications supported by Watson: PowerAqua [2].

PowerAqua is an open domain question answering system that exploits online semantic information. Exploiting the wealth of heterogeneous Semantic Web data now available is essentially about discovering interesting connections between items in a meaningful way. PowerAqua provides a natural language front end which makes it possible to perform question-answering on the Semantic Web, hence supporting such a discovery process. We show how PowerAqua makes use of the billion triple Watson to locate and extract answers to natural language questions from the provided datasets. As this "project" of using Watson and PowerAqua to find answers in the billion triple repository required a name, we chose to call it *FABilT: Finding Answers in a Billion Triples*.

The rest of the document is structured as follows. First, we describe Watson and report on the complex tasks of indexing the provided datasets and of deploying the Watson system to provide search and exploration functionalities over these datasets (Section 2). In Section 3, we give a quick overview of PowerAqua, highlighting in particular its ability to locate answers to natural laguage questions in large scale data spaces, by dynamically mapping user terminologies to ontological schemas. In this section we also report on the initial results obtained by trying out PowerAqua on the billion triples datasets. Finally, in Section 4, we conclude by detailing some of the lessons learned from this work.

2 Applying Watson to the One Billion Triples Challenge

For applications that wish to exploit the large amount of semantic information now available on the open web, there is a need for a new kind of infrastructure able to provide much more than the standard triple store solutions: a gateway to support them in locating, exploring and exploiting the data. This is indeed the goal of Watson: to provide all the necessary services so that semantic applications can be developed which can access data available on the Semantic Web in request to dynamic application needs –much like standard web search engines support dynamic access to web resources.

2.1 Overview of the Watson System

Watson makes use of a set of dedicated crawlers to collect online semantic content. A range of validation and analysis steps is then performed to extract several different types of information from the collected semantic documents, including: information about their content, their underlying knowledge representation languages, their relations with other documents, etc. This information provides a valuable base of metadata, exploited for indexing, searching and characterizing the content of the Semantic Web. Watson also provides a set of Web interfaces that allow human users to employ a variety of access mechanisms on the collected data (`http://watson.kmi.open.ac.uk`).

The core components of Watson are a set of Web Services and APIs which support the development of Semantic Web applications by providing various functionalities for searching and exploring online semantic documents, ontologies and entities within ontologies. The major functionalities concern:

- finding Semantic Web documents through flexible keyword based search.
- retrieving metadata (the size, language, label, logical complexity) about these documents, to help select the ones that are more suitable for a given scenario.
- finding specific entities (classes, properties, individuals) within a document.
- inspecting the content of a document, i.e., the semantic description of the entities it contains.
- applying SPARQL queries to individual Semantic Web documents, for example to get all the instances of a class.

The combination of mechanisms for searching semantic documents (keyword search), retrieving metadata about these documents and querying their content (e.g., through SPARQL) provides all the necessary elements enabling applications to select and exploit online semantic resources, without having to download the corresponding ontologies or to identify them at design time. In essence, Watson acts as a gateway to the Semantic Web as it provides applications with the ability to exploit the large scale, distributed body of knowledge that is dynamically evolving on the entire Semantic Web.

2.2 Indexing a Billion Triples in Watson

A major issue for a system like Watson when facing datasets like the billion triple ones concerns the off-line process of indexing this huge amount of data. Because Watson relies on the Lucene search engine library (`http://lucene.apache.org/`) to provide search and exploration features, we designed a complex process to create multiple Lucene indexes, each acting at a different "layer" of the semantic data (Ontologies, semantic documents, entities, relations, literals). As already discussed in the previous section, for each document in the datasets, this indexing process extracts useful information that is then used at run-time to search and navigate the data efficiently.

The scale of the provided datasets obviously represents a problem for such a process. Loading and parsing semantic data, processing them to extract information, and creating indexes for this information are all heavy tasks that require efficient computational mechanisms. Moreover, in addition to the scale of the data, their heterogeneity was a serious issue here. Indeed, our indexing process had to deal both with a very large overall number of documents (e.g., 6.6 million in the geonames dataset) and also with a small number of very big ones (e.g., up to 32GB in the dbpedia dataset).

To be able to handle very large numbers of documents, our process has been optimized and improved all along during the indexing of the billion triple datasets. At the moment, we are able to handle on average about 300,000 documents per day on a single computer (with 2 quad-core processors and 9GB of RAM). In addition, this process has been designed so that different documents can be indexed independently from each other, on different machines, producing separate sets of indexes that can be put together at run-time using a networked file system. This means that the indexing capability of Watson scales linearly with respect to the number of machines used. More precisely, we are now able to create complex multi-layer indexes for the entire collection of documents crawled by Swoogle (1.5 million documents, `http://swoogle.umbc.edu/`) in less than 5 days.[1]

The second big issue concerning indexing is that some datasets contained very large documents. Indeed, in our process, a document has to be loaded in memory, and an index for it has to be created before being dumped into a file. In addition, the performance of Lucene at run-time can decrease dramatically when such large documents are present in the index. To tackle this problem, we developed a method for partitioning documents prior to indexing. A number of smaller documents are created from a large one, which are indexed in such a way that the information about the provenance of the data they contain is kept in the indexes, so that, at run-time, they can be considered as part of the same document, like they were originally. Thanks to this technique, datasets such as dbpedia can be indexed in less than a day, and the indexing of large documents can also be distributed.

At the time of writing this submission, the major part of the datasets provided for the challenge have been indexed. However, due to limitations of the current hardware, not all of the datasets have been integrated in the final system yet. The current working system handles about 2.5 million documents corresponding to more than 9 million documents in the original dataset (due to the 6.6 million documents from geonames being grouped together in 100 documents). These 2.5 million documents correspond to more than 550 million triples. Within these datasets, Watson has extracted more than 80 million entities (classes, properties, individuals). The remaining datasets will be progressively added to the system by the time of the conference.

[1] Note however that, to facilitate their indexing, documents belonging to the geonames dataset have been grouped together, in 100 different groups, prior to indexing.

2.3 Accessing the Billion Triples through Watson

Interface: The standard Watson web interface (`http://watson.kmi.open.ac.uk`) has been reproduced to create a billion triple specific version, available at the following address: `http://kmi-web06.open.ac.uk:8081/WatsonWUI/`

This interface, is very similar to a Web search engine interface. A user can enter keywords to search for semantic documents that mention these terms. The result takes the form of a list of semantic documents (i.e., URIs) and, for each of them, the set of entities that match each of the given keywords is listed. Thanks to the complex metadata indexed by Watson, advanced search and navigation mechanisms can be used to explore the datasets and find relevant information. For example, the query can be restricted to the type of entities, scope and matching mechanism considered for the search. It is therefore possible to ask queries such as "Give me all the ontologies that contain classes or properties having an ID or a label matching exactly the keyword *author*."

Another important feature provided is that every URI, whether it refers to an entity or to a document, is clickable. For each document in the result, the link provided by its URI points to additional information about the document (such as its language, underlying description logic, imported ontologies, reviews provided by `http://revyu.com`, etc.) This page also gives access to additional functionalities on the particular document, like retrieving the original source, querying its content with SPARQL, or obtaining machine readable metadata about it in the OMV format.[2]

Going beyond the document level, Watson also allows users to explore the detailed content of the datasets. For each of the entities in the result, there is a link to a page providing the full semantic description of the entity for each of the documents it appears in. The provided information concerns the type of the entity (class, property, individual), the literals it is attached to (e.g. label, comment, etc.) and any relation it shares with any other entity of the collection (e.g. subclasses/superclasses, domain, range and other generic relations). Any related entity can be further inspected in the same way, allowing users to explore the content of the indexed documents by navigating from entity to entity.

API: As for the interface, the same Web services used for the standard version of Watson have been deployed for the billion triple version. Watson proposes a number of SOAP based services that contain all the functions necessary for applications to find, explore and exploit the collected documents. A lightweight Java API is also provided that acts as an interface between the application and the Watson services. The full documentation about these services and the corresponding API is available on the Watson website (`http://watson.kmi.open.ac.uk/WS_and_API.html`). The jar file containing the Watson API using the billion triple version of the services can be downloaded from (`http://kmi-web06.open.ac.uk:8081/watson-ws/watson-client-api-btc.jar`)

[2] SPARQL queries and OMV metadata are not yet available in the billion triple version.

Note that, while at the moment the billion triple version of Watson is kept separated from the standard Watson system, the two versions will be integrated in the near future, thus allowing applications and users to explore and use information seamlessly, regardless of whether it is provided by the billion triple datasets or other Semantic Web sources.

3 Exploring One Billion Triples with PowerAqua

3.1 Overview of PowerAqua

To some extent, PowerAqua can be seen as a straightforward human interface to any semantic document indexed by Watson. Using PowerAqua, a user can simply ask a question, like "Who are the members of the rock band Nirvana?" and obtain an answer, in this case in the form of a list of musicians (Kurt Cobain, Dave Grohl, Krist Novoselic and other former members of the group). The main strength of PowerAqua resides in the fact that this answer is derived dynamically from the relevant datasets available on the Semantic Web.

Without going into too many details, PowerAqua first uses a Gate-based [3] linguistic component to transform a question into a set of possible "query triples", such as <person/organization, members, rock band Nirvana>. The next step consists then in locating, thanks to Watson, online semantic documents describing entities that correspond to the terms of the query triples, locating for example an individual called *Nirvana* in a dataset about music. During this step, WordNet is used to augment the terms in the query triples with possible synonyms. Once a collection, usually rather large, of potential candidate ontologies is found, PowerAqua then employs a variety of heuristics and a powerful matching algorithm, PowerMap [4], to try and find answers from the collection of candidate ontologies. In our example, the query triple shown above can be successfully matched to the schema <Nirvana, has_members, ?x:Musician>, which has been found in a music ontology on the Semantic Web. In more complex examples, an answer may require integrating a number of statements. For instance, to answer a query such as "Which Russian rivers flow to the Black Sea", PowerAqua may need to find information about Russian rivers, information about rivers which flow to the Black Sea and then combine the two. In general, several sources of information, coming from various places on the Web, may provide overlapping or complementary answers. These are therefore ranked and merged according to PowerAqua's confidence in their contribution to the final answer.

3.2 Example

PowerAqua can handle a large variety of questions, of different forms, e.g., "Who are the members of the rock band Nirvana?", "Which members of KMi work on the Semantic Web?", "Give me the capitals of Europe", etc. To illustrate the issues faced by PowerAqua and its performance on the billion triple datasets, lets consider the query "find me rivers in Russia".

Thanks to Watson, PowerAqua is able to locate entities in various documents, which match the terms rivers and Russia, or their synonyms. In total, we get about 850 entities for rivers, and 570 entities for Russia. Then, PowerAqua explores each returned document in more detail, examining relations between the relevant entities. For example, for the document `http://www.aifb.uni-karlsruhe.de/WBS/meh/mapping/data/russia1a.rdf`, PowerAqua finds triples of the form <`http://lonely.org/russia#Russia`, `http://lonely.org/russia#has_major_river`, `http://lonely.org/russia#river`>, thus leading to the following results:

- `http://lonely.org/russia#Neva`.
- `http://lonely.org/russia#Don`.
- `http://lonely.org/russia#Dnepr`.
- `http://lonely.org/russia#Volga`.

A more interesting example is provided by the query "Which Russian rivers end in the Black Sea?". As already pointed out, PowerAqua is able to find mappings for the term "Russian river", however no triples covering the whole query can be found. Hence, PowerAqua tries again, after splitting the compound "Russian river", thus generating two separate query triples <russian, ?, rivers> and <russian / rivers, end, black sea>. Both these query triples find answers and the ones shared by both triples are returned, whether or not they originate from the same ontology. For instance, in `http://www.aifb.uni-karlsruhe.de/WBS/meh/mapping/data/russia1a.rdf`, the algorithm is able to find the triple <`http://lonely.org/russia#Dnepr`, `http://lonely.org/russia#flow_to`, `http://lonely.org/russia#Black_Sea`> that can be combined with the information shown earlier stating that the Dnepr is a Russian river to conclude that the Dnepr is a valid answer to the query. Because of the inherent redundancy in Semantic Web data, PowerAqua also finds confirmation that the Dnepr is indeed a Russian river from several other sources.

4 The Making of FABilT: Lessons Learned

One of the strengths of FABilT is that it considers both the infrastructure aspects (with Watson) and a concrete end-user application (with PowerAqua). We believe that this approach is essential to ensure that the proposed solutions do not only scale in theory, but are also usable and can be adopted by users in concrete scenarios. In addition, the experience of building the FABilT integrated system allows us to better understand the concrete issues and challenges faced when handling large scale data from the open Web from both points of view. Below we list some of the key issues which we have encountered when developing FABilT.

Large Scale + Heterogeneity = Big Trouble. An obvious characteristic of the billion triple repository is that it is very large. However, the really challenging aspect of these datasets appeared to be their heterogeneity. We already mentioned the difficulty of dealing with both very large documents and very

large numbers of them, when indexing in Watson. Analogously, coming up with generic solutions, working reliably and efficiently on datasets using different languages, and dealing with entities modeled at different levels of granularity and with different degrees of richness, are all challenges faced by both infrastructure components and applications that aim to make use of data from the Semantic Web.

Having Good Software to Rely on. The hardware resources employed for the development and deployment of FABilT are not enormous –mainly one reasonably sized server, with the occasional use of another one. While these resources are pushed to their limit, our experience shows that handling semantic information does not necessary require large data centers owned by well-resourced companies, but can still be handled within reasonably modest projects. This is both due to the availability of high quality software components (in our case Lucene and Jena, `http://jena.sourceforge.net/`) and to the focus we put in the development of Watson on providing comparatively high quality software.

Shifting Focus on Precision. Initial versions of PowerAqua were developed to work on local repositories or on an initial, sparsely populated Semantic Web. Therefore, the current PowerAqua algorithms have been designed to maximize recall (by augmenting the search space and including as much data as possible). As the billion triple data indexed by Watson provide efficient access to a huge amount of new semantic data, this approach becomes inappropriate, resulting in a slow overall performance and too much imprecision. Hence, new approaches need to be developed to automatically evaluate, rank and filter-out information coming from Semantic Web datasets, which must be able to prune the search space more drastically than the current version of PowerAqua. By doing this we will inevitably lower recall, hence some experimental evaluation will be needed to locate the optimal trade-off between recall and precision. In addition, as the size and heterogeneity of the sources considered by PowerAqua increases, trust becomes more and more important. For this reason, we are currently working on a novel trust engine, which will allow users of PowerAqua to provide feedback to the system, thus contributing to sophisticated models of the quality of information available in different Semantic Web sources.

References

1. d'Aquin, M., Baldassarre, C., Gridinoc, L., Sabou, M., Angeletou, S., Motta, E.: Watson: Supporting Next Generation Semantic Web Applications. In: Proc. of the WWW/Internet conference. (2007)
2. Lopez, V., Motta, E., Uren, V.: PowerAqua: Fishing the Semantic Web. In: Proc. of the European Semantic Web Conference, ESWC. (2006)
3. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics. (2002)
4. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Real Semantic Web on the Fly. In: Proc. of the 5th International Semantic Web Conference (ISWC). (2006)