Boom

info@boomamsterdam.nl
www.boomuitgeversamsterdam.nl

Iterative querying of the semantic web

*Reinoud Bosch & Ruben Verborgh**

Qualitative research is based on an iterative methodology (Bosch, this issue). Is it possible to develop algorithms for handling large amounts of data that are principally based on such iteration? This question will be addressed in this article in the context of using the so-called 'Semantic Web' as the source of large amounts of structured data for automatically answering questions. The 'Semantic Web' is the term that is used to refer to that part of the World Wide Web that can be automatically processed by machines (such as computers). It is constituted by so-called 'Linked Data': interlinked elementary 'triples' of information located in decentered networked systems of *de facto* databases (Berners-Lee et al., 2001; Berners-Lee, 2006). Triples are basic subject-predicate-object expressions, such as Oliver Sacks (subject) is (predicate) a scientist (object). To date, billions of such triples have been published on the public Web, spanning diverse domains such as life sciences, government, social networking, music and media, and encyclopedia knowledge.

In order to automatically process large amounts of triples of information to offer plausible answers to questions posed, software algorithms are required. The development of such algorithms has taken a huge flight, often inspired by correlation or the mimicking of the functioning of the human brain.[1] In contrast, in this paper, an approach is proposed in the form of an iterative mixed methods research cycle in which information is iteratively queried from the Semantic Web. The term 'querying' refers to the process of making a precise request for information retrieval from a website or database, such as asking for the names of scientists born in London. The task at hand is to see how and to what extent the iterative research cycle can be codified into algorithms with clear human control over the research process. As an indication of what such an iterative algorithm could look like in practice, a dynamic iterator pipeline algorithm is presented, the specifics of which will be described below. It is concluded that automatic iterative querying of large amounts of structured data is already occurring in practice, and that the challenge of broadening the scope of such iterative querying lies in the development of adequate metadata providing meaning to sets of triples, and the development of algorithm-based interfaces automatically applying logical iterative analytical steps. To advance both development processes, researchers need to describe iterative analytical steps to be coded into algorithms systematically and in detail.

* Dr. Ing. Reinoud Bosch is coordinating editor-in-chief of KWALON. Email: reinoudbosch@hotmail.com. Dr. Ruben Verborgh is a computer scientist, working as a postdoctoral researcher at Data Science Lab of Ghent University – iMinds, Belgium. Email: ruben.verborgh@ugent.be.

An iterative research cycle as the basis for algorithm development

What is the best way to go about answering research questions? On this, as methodologists are acutely aware, opinions vary widely. Different qualitative, quantitative, and mixed-methods approaches abound, in effect amounting to different views on what is the best approach to answering different questions in various situations. In practice, no consensus exists amongst researchers: some researchers have a tendency to believe that it is most appropriate to apply qualitative research methods, whereas others prefer quantitative methods. Such beliefs and preferences may depend on the topics at hand and the research questions posed, but also on philosophical and/or methodological convictions. Moreover, at the beginning of a research it is impossible to know whether a specific method chosen will in the end turn out to have been the best one that could have been chosen. The absence of consensus on, and the impossibility of *ex ante* certainty of, the best way to do research can be obviated by the use of an iterative research cycle that allows for the application of various qualitative, quantitative, and/or mixed-methods approaches according to the topics, research questions, convictions, and preferences of the researcher.

When applied to querying large amounts of structured data on the Semantic Web, such an iterative mixed-methods research cycle can be depicted in the form of a flowchart as in Figure 1. In a codified form that would allow for automatic processing, this iterative mixed methods research cycle would consist of an iterative data processing algorithm that goes through the logical steps of the posited research cycle. Presenting the research cycle in the form of a flowchart in principle allows for codification of the research cycle into algorithms by means of programming. As the flowchart indicates, at the beginning of the process a number of values need to be provided for the programming variables, research question, hypothesis, and sensitizing concepts/control variables. In the course of the process, these values are continuously adjusted and developed.

An important 'black box' in the research cycle is the component in which the results are analyzed (the box in Figure 1 that states 'Analyze results qualitatively, quantitatively, or both'). The analytical steps of this component depend on the topics and research questions at hand and the methodological convictions and preferences of the researcher. In the case of qualitative research, the steps of the analytical component will be iterative. Once the logic of the desired analytical steps has been clarified, an attempt can be made to codify those steps into algorithms. This could lead to a choice menu in which someone looking for an answer to a question can indicate which method(s) he or she believes to be best for that particular question in the situation at hand. The methods chosen may include such aspects as automatic coding in the case of qualitative research, automatic statistical analysis in the case of quantitative research, or approximate querying of the Semantic Web in a subsequent cycle.[2] It is important to be aware that the results that are retrieved through querying the Semantic Web are very basic units of information. This means that the way in which the Semantic Web is queried needs to be very specifically directed by the research question, the hypothesis, and the sensitizing con-

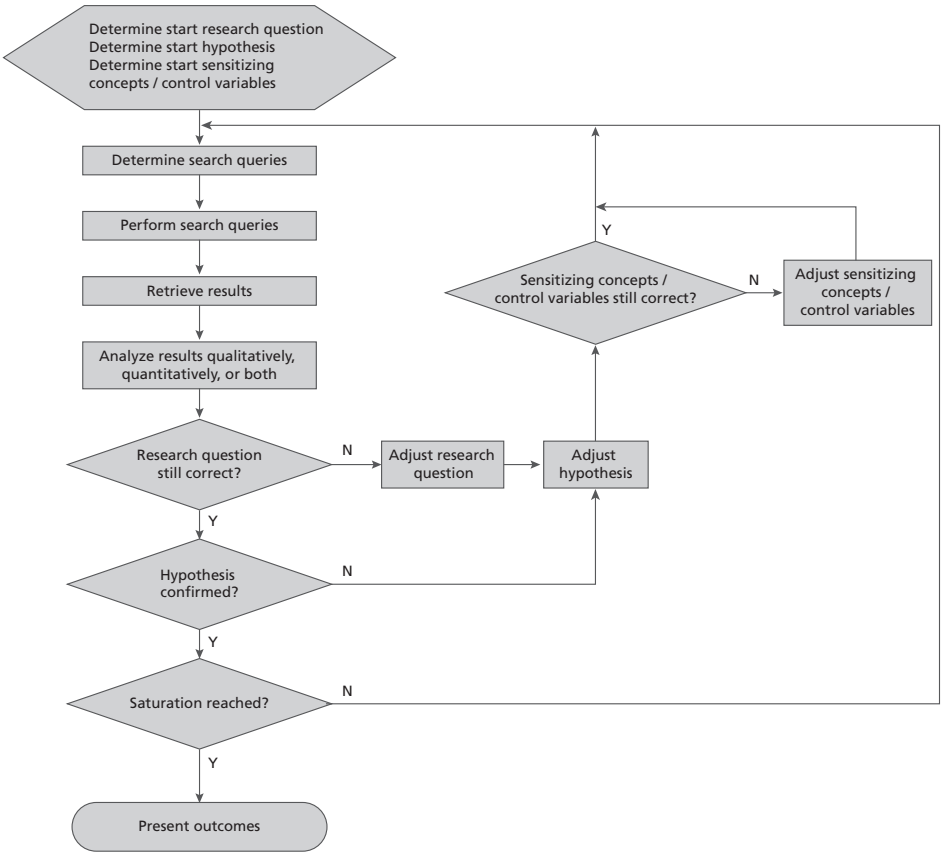

Figure 1 ***An iterative mixed methods research cycle for automatically querying the Semantic Web (based on Bosch, 2007, 2012, in press)***

cepts/control variables. This makes the iterative querying process explicitly a part of the overall research method – as depicted in Figure 1.

At this point in time, full codification of this iterative data processing algorithm is an ambitious project, especially if it should allow researchers to intervene in the process when deemed necessary. Only for quantitative approaches can iteration be implemented relatively straightforwardly. For qualitative and mixed methods research, the challenge lies in developing analytical algorithms that can offer a practical alternative to quantitative analysis. An indication of the type of analytical components that might be included in this project is provided by Friese (this issue). In the meantime, the purpose of depicting the research cycle is to provide a general overview of an automatic iterative approach to processing large amounts of structured data by means of querying the Semantic Web, and the relevance it accords to human control and intervention.

In practice, steps have already been taken in accordance with an automatic iterative research cycle for querying the Semantic Web – as in the dynamic iterator pipeline that has been developed for efficient and effective queries on the Semantic Web. This dynamic iterator pipeline will be described below. Taking this practical approach as a starting point, the question at hand becomes how to generalize this type of algorithm to the meta level – expanding it with logical components as described above – so as to encompass the entire research cycle.

Querying the Semantic Web

To demonstrate how querying the Semantic Web can be done in practice, this section will provide a simple example. To answer an actual research question, many different Semantic Web queries usually need to be combined and developed. As stated above, the Semantic Web consists of interconnected elementary triples of information – so-called RDF (Resource Description Framework) triples. These RDF triples consist of a subject, a predicate, and an object, as in "Oliver Sacks (subject) was born in (predicate) London (object)". An example of a semantic database that offers these kinds of triples is DBpedia, the RDF version of Wikipedia. Information on Oliver Sacks can be found on http://dbpedia.org/page/Oliver_Sacks.
To retrieve information from this type of database, the so-called SPARQL query language can be used, a language that allows for searching the Semantic Web in order to answer specific questions (Harris & Seaborne, 2013). As an example, the following SPARQL query aims to find a list of scientists born in London retrieved from dbpedia:

```
PREFIX dbpo: <http://dbpedia.org/ontology/>

SELECT ?person ?city WHERE {
    ?person rdf:type dbpo:Scientist.
    ?person dbpo:birthPlace ?city.
    ?city foaf:name "London"@en.
}
```

This query defines 'dbpo' as an abbreviation for the URI prefix (Uniform Resource Identifier – the name of a resource) 'http://dbpedia.org/ontology'. It consists of a basic graph pattern, which is the set of 3 triple patterns inside of the braces. The actual query for scientists born in London constructs a list of people (?person) who are (rdf:type) included in DBpedia ontology pages as scientists (dbpo:Scientist). The cities (?city) in which the selected people (?person) were born (dbpo:birthPlace) are also retrieved. Only those people are selected for which the city (?city) has the name (foaf:name) London ("London") in the English language (@en). When entered on the webpage http://dbpedia.org/sparql, a valid result would be

http://dbpedia.org/resource/Oliver_Sacks	http://dbpedia.org/resource/London

Oliver Sacks is a scientist, London is a city named "London", and Oliver Sacks was born there.
The problem, the solution of which the dynamic iterator pipeline contributes to, is that these queries cannot be reliably carried out on the Web. This is caused by the way in which Linked Data is currently published. The first way this is done is by the provision of archives that can be downloaded and queried locally. If a large part of or the entire Semantic Web is to serve as the relevant database to our questions, this would require downloading large parts or even the entire Semantic Web – which would defeat its purpose and be difficult and expensive. The other way is for RDF sources to provide a *query endpoint*, which accepts queries in the SPARQL language (such as http://dbpedia.org/sparql). This allows for the possibility to ask arbitrary questions about a dataset, but it imposes a large burden on the server that provides the SPARQL endpoint – as this server has to carry the computational burden of searching for and providing the requested information. In the first case, then, querying is slow and the client carries all the costs. In the second case, the server carries all the costs and querying is fast, but only if an endpoint for the dataset exists and is available – which is only the case for a minority of datasets. As a result, few datasets are reliably available for querying on the Web: most datasets exist only as dumps, and the relatively few public endpoints suffer from frequent downtime (Buil-Aranda et al., 2013).

Linked Data Fragments

To be able to understand the dynamic iterator pipeline that will be described below, the concept of Linked Data Fragment (LDF) needs to be described first (Verborgh et al., 2014). An LDF is a part of a Linked Data dataset relevant to a particular query. In the case of a data dump, the entire data dump is the fragment. In the case of a SPARQL endpoint, each possible query leads to a specific fragment. SPARQL queries can be made cost- and time-efficient by the use of an interface based on fragments that have a low generation cost for the server and a high information content for the SPARQL query. This can again be explained by means of a simple example concerned, in the case of the dynamic iterator pipeline, with finding a reliable and efficient way to perform Semantic Web queries. Answering a research question will require a totally different orientation and set of queries.
For purpose of understanding the dynamic iterator pipeline, suppose we want to find a reliable and cost and time efficient way of retrieving a list of scientists born in London from DBpedia given the basic graph pattern in the SPARQL query above. At the time of writing, DBpedia contains 20,695 scientists, 680,612 birthplace triples, and 19 things named 'London'. If this entire LDF were to be used by an interface in a query this would require processing 701,326 triples. But out of the 680,612 birthplace triples, only a minor fraction is about London. A more logical way to perform the query would be to retrieve the individual triple patterns of this basic graph pattern as follows:

- first retrieve all cities named London (matches: 19);
- for each such city, retrieve the people that were born there (matches: 8,210);

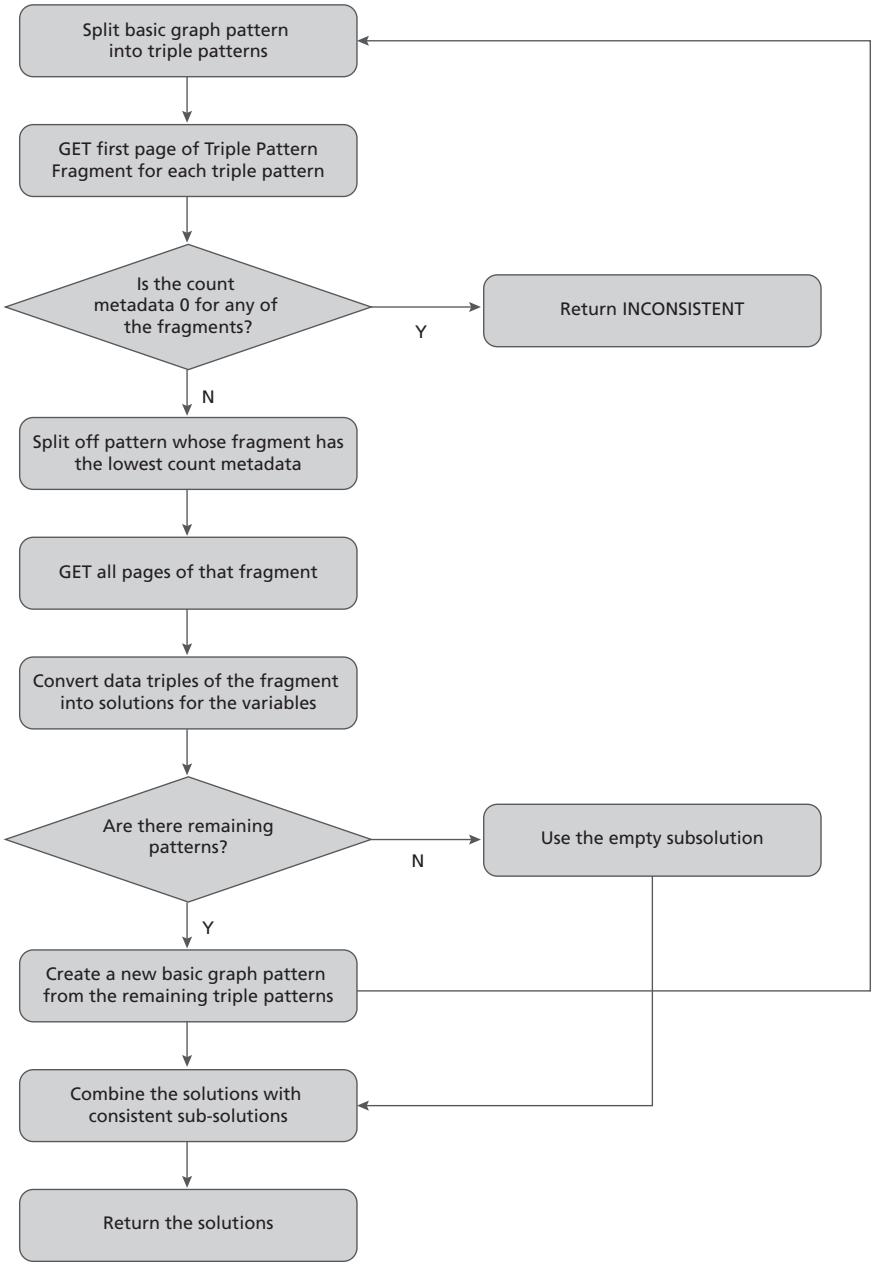

Figure 2 Flowchart of the dynamic iterator pipeline

- for each such person, verify whether he or she is a scientist (matches: 253).

This requires the processing of only 8,482 triples, or only 1.2% of the number used by the interface based on the entire LDF. This positive outcome is the result of human intervention, as we used our knowledge that there are fewer cities named "London" than people with a Wikipedia article mentioning their birthdate, and that only a minority of people in a city are scientists.
To some extent, such knowledge can be added to fragments as so-called 'metadata' – data that provide information about other data (Merriam-Webster, 2005). Concretely, a metadata triple may be added indicating the estimated total number of data triples in the fragment, similar to the 'number of search results' encountered in a Google search. For example, metadata could indicate that the fragment 'scientists' contains approximately 20,000, the fragment 'subjects and their birthplaces' approximately 680,000, and the fragment 'London' approximately 19 triples. When a query is executed, a client can use an interface that first peeks at the metadata and then starts with the fragment with the lowest estimated total number of data triples – in this case 'London' – after which the process repeats recursively.
An important practical problem remains. The server still has to generate the large fragment of 680,612 triples, regardless of whether the client will actually read it through the interface. To mitigate these problems, the server can *page* fragments. Each page then contains a certain number of triples (for instance, 100) together with the metadata triple. The final interface, consisting of paged triple pattern matches with count metadata, is called Triple Pattern Fragments (Verborgh et al., 2014).

The dynamic iterator pipeline

The dynamic iterator pipeline serves to perform the recursive process of the interface described above. The flowchart of the pipeline is presented in Figure 2. When a query for a basic graph pattern arrives, it is first split into triple patterns, because this is what the interface supports. Then, the first page of each of the fragments corresponding to these patterns are fetched, and their metadata is inspected. If any of those fragments is empty (metadata is zero), we know that no consistent solutions exist. Otherwise, we take the fragment with the most restrictive fragment (the one with the least number of matches) and retrieve all of its pages. The resulting data stream forms a partial solution. The remaining triple patterns undergo the process again, which yields another partial solution recursively. Both partial solutions are combined wherever this leads to consistent results, and the final solutions are returned.

Generalizing the logic of the dynamic iterator pipeline to the iterative research cycle

A clear analogy exists between the logic of the dynamic iterator pipeline and that of the iterative research cycle. Both start with applying analytical logic to a set of start values, which is followed by a Semantic Web query. The outcome of a subquery enters back into the pipeline/cycle, and analytical logic is applied to the merger of the results of the query and previous values of the programming variables. This iterative process continues until there are no remaining patterns/saturation is reached. In the dynamic iterator pipeline, metadata is added to existing Semantic Web triples indicating the number of triples; in the iterative research cycle, other types of metadata may be added providing meanings to sets of triples. An example of what such metadata may look like are the ontological categories being developed by the consortium of Bing, Google, Yahoo!, and Yandex in their schema.org initiative (http://schema.org/docs/full.html). The analytical logic that is selected together with the programming variables (start research question, hypothesis, sensitizing concepts/control variables) determines the modality of the interface through which the Semantic Web is queried. The dynamic iterator pipeline provides a practical and functioning model, the logic of which can be extended in the direction of a practically functioning iterative research cycle. The development of the logic of the iterative research cycle could be advanced by providing detailed and systematic answers to the question how researchers go about answering complex questions by combining information from different sources on the Web – which iteratively gets us back to where we started.

This is certainly an ambitious, complex, and challenging undertaking. But in practice, steps have already been taken, and with well-directed effort further steps toward development could be set. Otherwise, we are left with incomplete or misdirected efforts to the analysis of 'big data'.

Notes

1 See, e.g., Mayer-Schönberger & Cukier (2013); Frankish & Ramsey (2014); http://www.technologyreview.com/news/533686/2014-in-computing-breakthroughs-in-artificial-l-intelligence, consulted 2015-11-09.

2 See, e.g., Corby et al. (2006).

References

Berners-Lee, T. (2006). *Linked data*. http://www.w3.org/DesignIssues/LinkedData.html. Consulted: 2015-12-23.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001). The Semantic Web. *Scientific American*, *284*(5), 34-43.

Bosch, R. (2007). Pragmatism and the practical relevance of truth. *Foundations of Science*, *12*(3), 189-201.

Bosch, R. (2012). *Wetenschapsfilosofie voor kwalitatief onderzoek*. Den Haag: Boom Lemma.
Bosch, R. (this issue). Editorial: Qualitative research in the Digital Humanities. *KWALON 61*, *21*(1).
Bosch, R. (in press). *Power: A conceptual analysis*. The Hague: Eleven International Publishing.
Buil-Aranda, C., Hogan, A., Umbrich, J. & Vandenbussche, P.-Y. (2013). SPARQL Web-querying infrastructure: Ready for action? In *Proceedings of the 12th International Semantic Web Conference,* November.
Corby, O., Dieng-Kuntz, R., Gandon, F. & Faron-Zucker, C. (2006). Searching the Semantic Web: Approximate query processing based on ontologies. *IEEE Intelligent Systems & Their Applications*, *21*(1), 20-27.
Frankish, K. & Ramsey, W.M. (Eds.). (2014). *The Cambridge Handbook of Artificial Intelligence*. Cambridge: Cambridge University Press.
Friese, S. (this issue). Qualitative data analysis software – the state of the art. *KWALON 61*, *21*(1).
Mayer-Schönberger, V. & Cukier, K. (2013). *Big Data: A Revolution That Will Transform How We Live, Work and Think*. London: John Murray.
Merriam-Webster. (2005). *Merriam-Webster Collegiate Dictionary* (eleventh edition). Springfield, MA: Merriam-Webster.
Verborgh, R., Hartig, O., De Meester, B., Haesendonck, G., De Vocht, L., Vander Sande, M., Cyganiak, R., Colpaert, P., Mannens, E. & Van de Walle, R. (2014). Querying datasets on the Web with high availability. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandečić, P. Groth, N. Noy, K. Janowicz & C. Goble (Eds.), *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I* (pp. 180-196). Berlin: Springer.