Towards LarKC: a Platform for Web-scale Reasoning

Dieter Fensel (University of Innsbruck) Frank van Harmelen (Vrije Universiteit Amsterdam)
Bo Andersson (Astrazeneca AB) Paul Brennan (International Agency for Research on Cancer)
Hamish Cunningham (University of Sheffield) Emanuele Della Valle (CEFRIEL)
Florian Fischer (University of Innsbruck) Zhisheng Huang (Vrije Universiteit Amsterdam)
Atanas Kiryakov (OntoText Lab, Sirma AI Ltd.) Tony Kyung-il Lee (Saltlux)
Lael Schooler (Max Planck Institute for Human Development, Berlin) Volker Tresp (Siemens)
Stefan Wesner (University of Stuttgart) Michael Witbrock (Cycorp Europe)
Ning Zhong (Beijing University of Technology)

Abstract

Current Semantic Web reasoning systems do not scale to the requirements of their hottest applications, such as analyzing data from millions of mobile devices, dealing with terabytes of scientific data, and content management in enterprises with thousands of knowledge workers. In this paper, we present our plan of building the Large Knowledge Collider, a platform for massive distributed incomplete reasoning that will remove these scalability barriers. This is achieved by (i) enriching the current logic-based Semantic Web reasoning methods, (ii) employing cognitively inspired approaches and techniques, and (iii) building a distributed reasoning platform and realizing it both on a high-performance computing cluster and via ”computing at home”. In this paper, we will discuss how the technologies of LarKC would move beyond the state-of-the-art of Web-scale reasoning.

1. Introduction

Michael Lynch, CEO and Founder of Autonomy[1], recently stated that ”meaning-based computing is the way of the future as 80 per cent of information within enterprises is unstructured and that understanding this ’hidden’ intelligence is at the heart of improving the way we interact with information”. Some of the most advanced use cases for such semantic computing today require reasoning about 10 billion RDF triples in less than 100 ms. These numbers originate from the telecom sector aiming to generate revenue streams through new context-sensitive and personalized mobile services, but this is just one example of a general demand. The Web has made tremendous amounts of information available that could be processed based on formal semantics attached to it. The Semantic Web has developed a number of languages that use logic for this purpose. However, current logic based reasoning systems does not scale to the amount of information and the setting that is required for the Web. The inherent trade-off between the expressiveness of a logical representation language and scalability of reasoning over the information has been clearly observed from a theoretical point of view [2] but has also show to have a very practical impact on possible use-cases [9].

Thus a reasoning infrastructure must be designed and built that can scale and that can be flexibly adapted to the varying requirements of quality and scale of different use-cases. If such an infrastructure is not built, ”meaning-based computing” will never happen on the Web and will remain confined to well-controlled data-sets inside company intranets.

In this paper, we present our plan of building the Large Knowledge Collider, a platform for Web-scale reasoning, which research is carried on by the EU 7th framework Project LarKC[2]. The aim of LarKC is to develop a platform for massive distributed incomplete reasoning that will remove the scalability barriers of currently existing reasoning systems for the Semantic Web. This will be achieved by:

- Enriching the current logic-based Semantic Web reasoning with methods from information retrieval, machine learning, information theory, databases, and probabilistic reasoning.
- Employing cognitively inspired approaches and techniques such as spreading activation, focus of attention, reinforcement, habituation, relevance reasoning, and bounded rationality.
- Building a distributed reasoning platform and realizing it both on a high-performance computing cluster and via ”computing at home”.

[1] Europe’s second largest software company, with enterprise search and knowledge management as its core markets.

[2] http://www.larkc.eu

978-0-7695-3279-0/08 $25.00 © 2008 IEEE
DOI 10.1109/ICSC.2008.41

The rest of the paper is organized as follows. Section 2 presents the visions, missions, objectives and strategies of LarKC as an overview of the Large Knowledge Collider. Section 3 discusses the state-of-the-art of Web-scale reasoning and its relations with LarKC. Section 4 provides several use cases of LarKC. Section 5 concludes the paper.

2. Overview of LarKC

2.1. Vision

The driving vision behind LarKC is to go beyond the limited storage, querying and inference technology currently available for semantic computing. The fundamental assumption taken is that such an infrastructure must go beyond the current paradigms which are strictly based on logic. By fusing reasoning (in the sense of logic) with search (in the sense of information retrieval), and taking seriously the notion of limited rationality (in the sense of Herbert Simon [11]) we will obtain the paradigm shift that is required for reasoning at Web scale.

The overall vision of LarKC is: To build an integrated platform for semantic computing on a scale well beyond what is currently possible. The platform will fulfill needs in sectors that are dependent on massive heterogeneous information sources such as telecommunication services, biomedical research, and drug-discovery. The platform will have a pluggable architecture in which it is possible to exploit techniques and heuristics from diverse areas such as databases, machine learning, cognitive science, Semantic Web, and others. The platform will be implemented on a computing cluster and via ”computing at home”, and will be available to researchers and practitioners from outside the consortium to run their own experiments and applications, using suitable plug-in components.

2.2. Missions and Objectives

We will develop the Large Knowledge Collider as a pluggable algorithmic framework which will be implemented on a distributed computational platform. It will scale to web-reasoning by trading quality for computational cost and by embracing incompleteness and unsoundness.

Pluggable: Instead of being built only on logic, the Large Knowledge Collider will exploit a large variety of methods from other fields: cognitive science (human heuristics), economics (limited rationality and cost/benefit trade-offs), information retrieval (recall/precision trade-offs), and databases (very large datasets). A pluggable architecture will ensure that computational methods from these different fields can be coherently integrated.

Distributed: The Large Knowledge Collider will be implemented on parallel hardware using cluster computing techniques, and will be engineered to be ultimately scalable to very large distributed computational resources, using techniques like those known from SETI@home.

The LarKC major objectives are:

- Design an integrated pluggable platform for large-scale semantic computing.
- Construct a reference implementation for such an integrated platform for large-scale semantic computing, including a fully functional set of baseline plug-ins.
- Achieve sufficient conceptual integration between approaches of heterogeneous fields (logical inference, databases, machine learning, cognitive science) to enable the seamless integration of components based on methods from these diverse fields.
- Demonstrate the effectiveness of the reference implementation through applications in services based on data-aggregation from mobile-phone users, meta-analysis of scientific literature in cancer research, and data-integration and -analysis in early clinical development and the drug-discovery pipeline.

2.3. Strategies

The Large Knowledge Collider will consist of a number of pluggable components: retrieval, abstraction, selection, reasoning and deciding. These components are combined in a simple algorithmic schema, which is shown in Algorithm 1. Researchers can design and experiment with multiple realizations for each of these components.

Algorithm 1 Algorithmic Schema

```
loop
    Obtain a selection of data (RETRIEVAL)
    transform to an appropriate representation (ABSTRACTION)
    draw a sample (SELECTION)
    reason on the sample (REASONING)
    if more time is available
    and/or the result is not good enough (DECIDING) then
        increase the sample size (RETRIEVAL)
    else
        exit
    end if
end loop
```

In LarKC, massive, distributed and necessarily incomplete reasoning is performed over web-scale knowledge sources. Massive inference is achieved by distributing problems across heterogeneous computing resources and coordinated by the LarKC platform. This overall architecture is depicted in Figure 1. Some of the distributed computational resources will run highly coupled, high performance inference on local parallel hardware before communicating results back to the distributed computation. With Web-scale inference, complete information is an empty hope and the distributed computation shown in the left part of Figure 1 include some failed computations that do not thwart the entire problem solving task. The right side of the figure illustrates the architecture that achieves this. LarKC allocates resources strategically and tactically, according to its basic algorithmic schema, to: 1) Retrieve raw content and assertions that may contribute to a solution, 2) Abstract that information into the forms needed by its heterogeneous reasoning methods, 3) Select the most promising approaches to try first, 4) Reason, using multiple deductive, inductive, abductive, and probabilistic means to move closer to a solution given the selected methods and data, and 5) Decide whether sufficiently many, sufficiently accurate and precise solutions have been found, and, if not, whether it's worth trying harder. This basic problem framework is supplied as a plug-in architecture, allowing intra-consortium and extra-consortium researchers and users to experiment with improvements to automated reasoning.

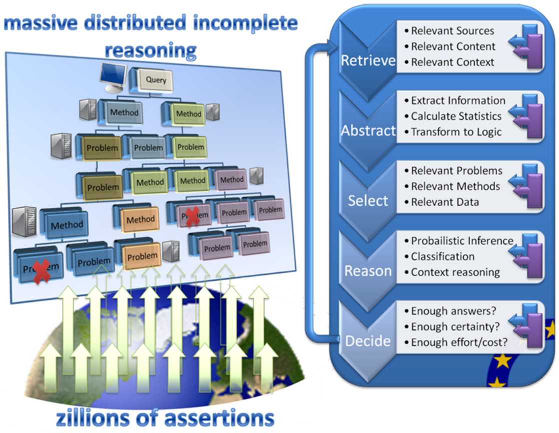

Figure 1: LarKC Inference Strategy and Architecture

This architecture will enable the productive yet frictionless interaction between components and the various disciplines related with them, which are shown in Table 1.

Plugin Component	Based on results in...
RETRIEVAL	Information Retrieval, Cognition, ...
ABSTRACTION	Machine Learning, Ontology,...
SELECTION	Statistics, Machine Learning, Cognition, ...
REASONING	Logic, Probabilistic inference, ...
DECIDING	Economics, Computing, Decision theory...

Table 1: Disciplines

3. State-of-the-art in Relevant Areas

3.1. Reasoning

Researchers have developed methods for reasoning in rather small, closed, trustworthy, consistent, and static domains. They usually provide a small set of axioms and facts. DL reasoners can deal with 10^5 axioms (concept definitions), but they scale poorly for large instance sets. Logic programming engines can deal with similar-sized rule sets as well as larger instance sets (say, 10^6), but they can draw only simple logical conclusions from these theories. Both streams are highly interesting areas of research, and topics such as how to combine them attract a lot of attention (i.e. [5], [3]).

Still, there is a deep mismatch between reasoning on a Web scale and efficient reasoning algorithms over restricted subsets of first-order logic. This is rooted in underlying assumptions of current systems for computational logic:

- Small set of axioms. If the Web is to capture the entirety of human knowledge, the number of axioms ends up being very large.
- Small number of facts. Assuming a Google count of roughly 30 billion Web pages and a modest estimate of 100 facts per page, we consequently are in the order of a trillion facts.
- Completeness of inference rules. The Web is open, with no defined boundaries. Therefore, completeness is an unrealistic requirement for an inference procedure in this context.
- Correctness of inference rules and consistency. Traditional logic takes axioms as reflecting truth and tries to infer the implicit knowledge they provide (their deductive closure). In a Web context, information is unreliable from the beginning, which means even a correct inference engine cannot maintain truth.
- Static domains. The Web is a dynamic entity: the known facts will change during the process of acquiring and using them.

Each of these assumptions needs to be revisited and adapted to the reality of the web.

Merging reasoning and Web-search: The only way to bridge the divide between reasoning and the Web is to interweave the reasoning process with the process of establishing the relevant facts and axioms through retrieval (ranking or selection) and abstraction (compressing or transforming information). In this way, retrieval and reasoning become two sides of the same coin – a process that aims for useful information derived from data on the Web.

From completeness and soundness to recall and precision: The project will fill a significant gap between the current methods for indexing and searching on the one hand, and reasoning on the other hand. To illustrate this gap, let's consider the typical success measures used by the searching and the reasoning communities. The conventional method for measuring success in search task is in terms of the levels of precision and recall of the results. It is well known that higher precision leads to lower recall, and vice versa.

Turning to reasoning, the majority of previous work has considered processes that are sound and complete, which, in an information seeking context, equates to 100% precision and 100% recall. In practice this type of perfection has both high setup costs (codifying domain expertise in an ontology and semantically structuring the entire information space) and of computation. The Large Knowledge Collider will exploit the space of possibilities above the precision/recall curve but below perfect completeness and soundness, using methods that trade off these properties of logic in exchange for lower costs of creation and computation. Pictorially, the project positions itself in the space between traditional search and traditional reasoning.

Feature-based comparison with existing state of the art reasoning platforms is shown in Table 2.

Feature	SoA	LarKC
Reasoning method	Logic only	Multiple methods
Heuristics	Hardwired	Configurable plugins
Precision/recall	100/100	Configurable trade-off
Dynamic axioms and facts	Not supported	Supported
RDF retrieval scale	$O(10^9)$	$O(10^{12})$
RDF inference scale	$O(10^8)$	$O(10^{10})$
Anytime behavior	None	Configurable
Parallelism	None	local cluster $O(10^2)$ machines wide-area distribution $O(10^3)$ machines

Table 2: Feature-based comparison with existing state of the art reasoning platforms.

3.2. Scalable Semantic Repositories

Ontotext produces high-performance RDF- and OWL-storage and inference technology and actively monitors the state-of-the-art in this area. [10] provides extensive performance figures which we summarize in Table 3; it provides a comparison of the tools in terms of scalability, speed, and inference capabilities. The Cycorp inference engine [8], which is able to deal with knowledge bases containing several millions of assertions, will be used complementary to Ontotext technology with respect to reasoning tasks.

All the figures refer to the best published results for the corresponding tools. Direct comparisons of the tools should be made with caution. As with the relational DBMS, an engine's performance can vary considerably depending on the configuration and tuning with respect to the specific task or benchmark. Semantic repositories are even harder to compare because they also perform inference, which can be implemented in a wide variety of modalities - for example, forward-chaining harms the loading performance, while backward-chaining slows down the query evaluation. The storage and querying functionality of the Large Knowledge Collider will be based on the Ontotext technology behind SwiftOWLIM and BigOWLIM [7] and on related technology provided by CycEur. Consequently we are indeed building on top of the best that the current state-of-the-art has to offer.

Tool	Scale (mil. of statem.)	Inference	Load Speed in 1000 st./sec	Hardware (GB RAM)
KAON2	˜ 10	OWL DL + rules	20	0.5
RacerPro	1	OWL DL	-	0.5
Minerva (IBM)	2	OWL Lite +/-	>1	0.5
Triple20	40	OWL Lite +/-	6	2
SwiftOWLIM	10-80	OWL Lite +/-	20-60	1-16
Sesame 2.0 NS	70	RDFS +	6	0.8
ORACLE10R2	100	RDFS +	>1	2
Jena v2.1/2.3	7-200	-	?-6	2-?
KOWARI	235	None	4	?
RDFGateway	262	OWL Lite +/-	>1	?
AlegroGraph	1 000	RDFS -	20	2
OpenLink	1 060	None	12	8
BigOWLIM	1 060	OWL Lite +/-	4	12

Table 3: Scalable Semantic Repositories

In a very recent development, DERI Galway has implemented a distributed RDF store which reported real-time performance with up to 7 billion triples [6]. This system is limited to only retrieving statements, and does not perform any inference. Clearly, this impressive technology is an excellent basis for implementing basic indexing and lookup techniques, but is by itself not enough. The essential difference here is similar to that between a database and a heuristic inference system. The Large Knowledge Collider is aiming at inference, not only at lookup, with search as its major paradigm, as opposed to retrieval. Clearly no amount of improvement on efficient indexing systems will be able to beat the exponential search-space generated by inference.

The Large Knowledge Collider will of course exploit such state-of-the-art indexing systems but is aiming deliberately for imperfect recall and precision as means to achieve unlimited scalability (through exchanging quality for performance), while maintaining task-related bounded rationality.

3.3. Web Search

Information retrieval (IR) technology has proliferated in rough proportion to the expansion of knowledge and information as a central factor in economic success. The main dimensions conditioning choice of search technology are:

Volume. The GYM big three search engines (Google, Yahoo!, Microsoft) deliver sub-second responses to hundreds of millions of queries daily over hundreds of terabytes of data. At the other end of the scale desktop search systems can rely on substantial computing resources relative to a small data set.

Value. The retrieval of high-value content (typically within corporate intranets or behind pay-for-use turnstiles) is often mission-critical for the organization that owns the content. For example the BBC allocates a skilled staff member for eight hours per broadcast hour to index their most important content.

To process web-scale volumes GYM use a combination of one of the oldest and simplest retrieval data structures (an inverted file that relates search terms to documents) and a ranking algorithm whose most important component is derived from the link structure of the Web. In general, when specifying a search, users enter a small number of terms as a query, based on words that people expect to occur in the types of document they seek. This gives rise to a fundamental problem, known as ”index term synonymy”: not all documents will use the same words to refer to the same concept. Therefore, not all the documents that discuss the concept will be retrieved by a simple keyword-based search. Furthermore, query terms may of course have multiple meanings; this problem is called ”query term polysemy”. The ambiguity of the query can lead to retrieval of irrelevant information and/or failure to retrieve relevant information. High-value (or low-volume) content retrieval systems address these problems with a variety of semantics-based approaches that attempt to perform conceptual indexing and logical querying. For example, the BBC system mentioned previously performs indexing using a thesaurus of 100,000 terms that generalizes over anticipated search terms. Life Sciences publication databases increasingly use rich terminological resources to support conceptual navigation (MeSH, the Gene Ontology, Snomed, the unified UMLS system, etc). Our task in LarKC is to show how the high-value techniques can scale to higher volumes than is currently feasible.

3.4. Semantic Spaces

Data spaces are shared virtual data spaces which are designed for concurrent access by multiple processes. The data published and exchanged in the space is generally expressed as an atomic unit, called a tuple, consisting of typed fields. Processes interact with spaces through a minimal language or an interface which expresses a coordination model with concurrency. The idea of considering space-based communication for the Semantic Web has led to the vision of Triple Spaces [4], middleware which allows shared access to semantic information in a structured and reliable way. Triple Spaces emphasizes the use of the space for the publication and retrieval of ”payload” data that represent the know-how published on the Web. The structured representation of RDF triples in a space allows for semantic reasoning about this data, whilst the space guarantees the reliable and efficient placement of this knowledge, and additionally provides a smart interface including near-time notification about changes.

A first European project on semantic spaces has been initiated in April 2006. The project TripCom provides the foundational conceptual work for semantic spaces and a prototypical implementation is expected for 2008. However, as the first project in this field, TripCom uses primarily RDF as a representation language for space data, and exploits reasoning on semantic data only to a limited extent. The latter is also due to the the lack of appropriate instruments to combine reasoning with the coordination model underlying such systems.

A first step in advancing the state-of-the-art will thus have to aim at taking up the ideas towards fully Semantic Web-enabled spaces and to design and implement the extensions or changes needed in order to feasibly support also more expressive Semantic Web formalisms, and to embed reasoning into distributed spaces. Further on, as a distributed space middleware will be available, the semantics of the underlying coordination model (e.g. the meaning of the operations for publishing and retrieving data) will need to be rethought in order to adapt to the particularities of an open globally networked environment such as the Web, in which completeness and correctness can not be guaranteed. With this respect, results of the LarKC project could contribute to the further development of space-based technology.

4. Use Cases

4.1. Real Time City

One of the major technical problems that hinder the development of real time control to cities is the difficulty in answering massive numbers of queries. These queries requiring reasoning about huge streams of facts, originating in real time from heterogeneous and noisy data sources, in order to control traffic flow to help citizens get where they need to be. For instance, consider that 30-40% of car fuel in large cities is spent by drivers in looking for a parking lot and this share dramatically increases when big events involving lots of people take place. Image if it were possible to predict potential congestion problems, and help people avoid the congestion to ensure that they reach the event in time.

A large amount of information is already available at almost no cost: all the commercial activities and meeting places (cf. Google Earth), all events scheduled in the city and their location, all the position and speed information from public transportation vehicles and mobile phone users, and the availability of all free parking lots in specific parking areas and their position. For a large city it is not hard to imagine that there could be billions of triples that are continuously being updated.

A reasoner of new generation is clearly needed in order to infer for each citizen, who wants to attend an event, the most convenient parking area, a place to meet with friends, the fastest route to such place to instruct the car GPS. And time constrains for such a reasoner are very demanding (i.e., few ms per query) if they take into consideration that citizens are free to follow or discard the suggestions proposed by the city real time control system, therefore continuous inference is required.

4.2. Data Integration for Early Clinical Drug Development

The number of different databases used by life-science researchers, as well as the diversity of their schemata and intrinsic semantics, makes semantic data integration extremely relevant to this domain.

Data integration is one of the most challenging, expensive and pressing IT problem tackled by the pharmaceutical companies in their drug development process. In the recent years, there is an increasing generation of pharmaceutical data as result of the R&D process. With high-throughput technologies it is possible to analyze thousands of genes and gene products in a very short time. Despite the high volume of newly discovered biological data, the pharmaceutical companies have so far not managed to integrate this patient, disease, efficacy and safety data in a way to meet their expectations of increased productivity and pipeline throughput. In 2005, there were 719 publicly available databases, 171 more than in 2004. These exhibit extreme heterogeneity of information type and there is no single database that supplies a complete picture for decision making in drug discovery and development. Most of the databases are developed independently and address local and specific needs.

The volume of the data, the degree of interconnection and the complexity of the reasoning necessary for the integration, puts the problem far beyond the capabilities of contemporary inference engines.

There are very few engines which can load the OWL variant of the UNIPROT database and reason on top of it. Even those which can, suffer performance which is impractical for most applications. Integration with other databases or with experimental results is simply unthinkable at present. we aim for semantic integration of life sciences databases into an ontology-structured knowledge base and annotation of scientific papers both post-hoc and during authoring with respect to the knowledge base. These scenarios will pose heavy loads on LarKC components and thus test their performance in practice.

5. Conclusion

With the rapid growth of available data and knowledge in standardised Semantic Web formats, there is a pressing need for a reasoning platform that can handle very large volumes of such data.

Furthermore the Web is an architecture of standards and formalisms for many heterogeneous applications, fulfilling very different demands. And while this multiple-facets are one of the very building blocks of the Web they also imply that basic assumptions of reliability, trustworthiness, and consistency not necessarily hold. Moreover, as the Web is a democratic and open medium in which “publishing is cheap” [1], we should even expect inconsistency (due to malicious content or simply disagreement about a state of things).

LarKC aims to be the platform to address these issues, and is built on the following principles:

- Achieve scalability through *parallelisation*. Different possibilities are offered either through tight integration of parallel processes on cluster-style hardware, or through much looser coupled wide-area distributed computing.
- Achieve scalability through *giving up completeness*. Partial reasoning results are useful in many domains of application. Significant speedups and can be obtained by incompleteness in many stages of the reasoning process, ranging from selection of the axioms to incomplete reasoning over those axioms.
- Do not build a single reasoning engine that is supposed to be suited for all kinds of use-cases, but instead build a *configurable platform* on which different components can be plugged in to obtain different scale/efficiency trade-offs, as required by different use-cases.

References

[1] T. Berners-Lee, W. Hall, J. Hendler, K. O’Hara, N. Shadbolt, and D. Weitzner. A framework for web science. *Foundations and Trends in Web Science*, 1(1):1–130, 2006.

[2] R. Brachman and H. Levesque. The tractability of subsumption in frame-based description languages. *Proc. of the 4th Nat. Conf. on Artificial Intelligence (AAAI-84)*, pages 34–37, 1984.

[3] F. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. AL-log: Integrating Datalog and Description Logics. *Journal of Intelligent Information Systems*, 10(3):227–252, 1998.

[4] D. Fensel. Triple-space computing: Semantic Web Services based on persistent publication of information. *Proceedings of the IFIP International Conference on Intelligence in Communication Systems*, 2004.

[5] B. Grosof, I. Horrocks, and R. Volz. Description logic programs: combining logic programs with description logic. *Proceedings of the 12th international conference on World Wide Web*, pages 48–57, 2003.

[6] A. Harth, J. Umbrich, A. Hogan, and S. Decker. Yars2: A federated repository for querying graph structured data from the web. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, volume 4825 of *LNCS*, pages 211–224. Springer Verlag, November 2007.

[7] A. Kiryakov, D. Ognyanov, and D. Manov. OWLIM–a Pragmatic Semantic Repository for OWL. *WISE Workshops*, pages 182–192, 2005.

[8] D. Lenat. CYC: a large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[9] M. Luther, S. Bohm, M. Wagner, and J. Koolwaaij. Enhanced presence tracking for mobile applications. *Proc. of ISWC*, 5, 2005.

[10] D. Ognyanoff, A. Kiryakov, R. Velkov, and M. Yankova. D2. 6.3 A scalable repository for massive semantic annotation. *Technical report, SEKT*, 2007.

[11] H. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69(1):99–118, 1955.