Distributed Semantic Analytics using the SANSA Stack

Jens Lehmann[1,2], Gezim Sejdiu[1], Lorenz Bühmann[3], Patrick Westphal[3], Claus Stadler[3], Ivan Ermilov[3], Simon Bin[3], Nilesh Chakraborty[1], Muhammad Saleem[3], Axel-Cyrille Ngonga Ngomo[3,4], and Hajira Jabeen[1]

[1] University of Bonn, Germany
{jens.lehmann, sejdiu, chakrabo, jabeen}@cs.uni-bonn.de
[2] Fraunhofer IAIS, Bonn, Germany
jens.lehmann@iais.fraunhofer.de
[3] Institute for Applied Informatics (InfAI), University of Leipzig, Germany
{buehmann,patrick.westphal,cstadler,iermilov,sbin,saleem}@informatik.uni-leipzig.de
[4] Paderborn University, Data Science Group, Paderborn, Germany
axel.ngonga@uni-paderborn.de

Resource type Software Framework
Website `http://sansa-stack.net`
Permanent URL `https://figshare.com/projects/SANSA/21410`

Abstract. A major research challenge is to perform scalable analysis of large-scale knowledge graphs to facilitate applications like link prediction, knowledge base completion and reasoning. Analytics methods which exploit expressive structures usually do not scale well to very large knowledge bases, and most analytics approaches which do scale horizontally (i.e., can be executed in a distributed environment) work on simple feature-vector-based input. This software framework paper describes the ongoing Semantic Analytics Stack (SANSA) project, which supports expressive and scalable semantic analytics by providing functionality for distributed computing on RDF data.

1 Introduction

In this paper, we introduce SANSA,[5] an open-source[6] *structured data processing engine* for performing distributed computation over large-scale RDF datasets. It provides data distribution, scalability, and fault tolerance for manipulating large RDF datasets, and facilitates analytics on the data at scale by making use of cluster-based big data processing engines. It comes with: (i) specialised serialisation mechanisms and partitioning schemata for RDF, using vertical partitioning strategies, (ii) a scalable query engine for large RDF datasets and different distributed representation formats for RDF, namely graphs, tables and tensors, (iii) an adaptive reasoning engine which derives an efficient execution and evaluation plan from a given set of inference rules, (iv) several distributed structured machine learning algorithms that can be applied on large-scale RDF data, and (v) a

[5] `http://sansa-stack.net/`
[6] `https://github.com/SANSA-Stack`

framework with a unified API that aims to combine distributed in-memory computation technology with semantic technologies.

To achieve the goal of storing and manipulating large RDF datasets, we leverage existing big data frameworks like Apache Spark[7] and Apache Flink,[8] which have matured over the years and offer a proven and reliable method for general-purpose processing of large-scale data.

The remainder of the paper is structured as follows: section 2 depicts a new vision of combining distributed computing frameworks with the semantic technology stack and an overview of the SANSA architecture. We present some of the use cases demonstrating a variety of applications of the SANSA framework in detail in section 3. We discuss related work in section 4 and conclude in section 5 along with directions for future work.

2 Vision and Architecture

Research efforts in the areas of distributed analytics and semantic technologies have so far been mostly isolated. As illustrated in Figure 1, we see several core aspects in which both areas have complementary strengths and weaknesses.

State-of-the-art distributed in-memory analytics frameworks, such as Apache Spark and Apache Flink, provide graph-based analytics [1] but do not support semantic technology standards. The application of these approaches on heterogeneous data sources faces many limitations, in particular due to non-standardised input formats and the need for manual data integration. This can lead to large amounts of time and effort being spent on pre-processing data rather than performing the actual data analytics task. Semantic technologies are W3C-standardised and have the potential to significantly alleviate the pre-processing overhead: although the initial effort for modelling input data in RDF may be higher, the repeated reuse of the datasets in various analytics tasks can lead to a reduction of overall effort. Moreover, there are many connectors from existing data sources to RDF (e.g. via the R2RML standard) and they provide sophisticated data integration, e.g. via link discovery and fusion approaches for RDF. We want to go a step further and use this modelling standard as a basis for machine learning and data analytics. The layered architecture of SANSA is a direct consequence of this vision and is depicted at the top of Figure 1. We will now discuss the different layers and currently implemented functionality in SANSA.

Knowledge Distribution & Representation Layer[9][10] This is the lowest layer on top of the existing distributed frameworks (Apache Spark or Apache Flink). It provides APIs to load/store native RDF or OWL data from HDFS or a local drive into the framework-specific data structures, and provides the functionality to perform simple and distributed manipulations on the data. Moreover, it allows the users to compute RDF statistics described in [7] in a distributed manner. For the representation of OWL axioms, we are also investigating data structures that allow an efficient, distributed computation of light-weight reasoning tasks like inferring the closure w.r.t. sub class relations.

[7] http://spark.apache.org/

[8] http://flink.apache.org/

[9] https://github.com/SANSA-Stack/SANSA-RDF

[10] https://github.com/SANSA-Stack/SANSA-OWL

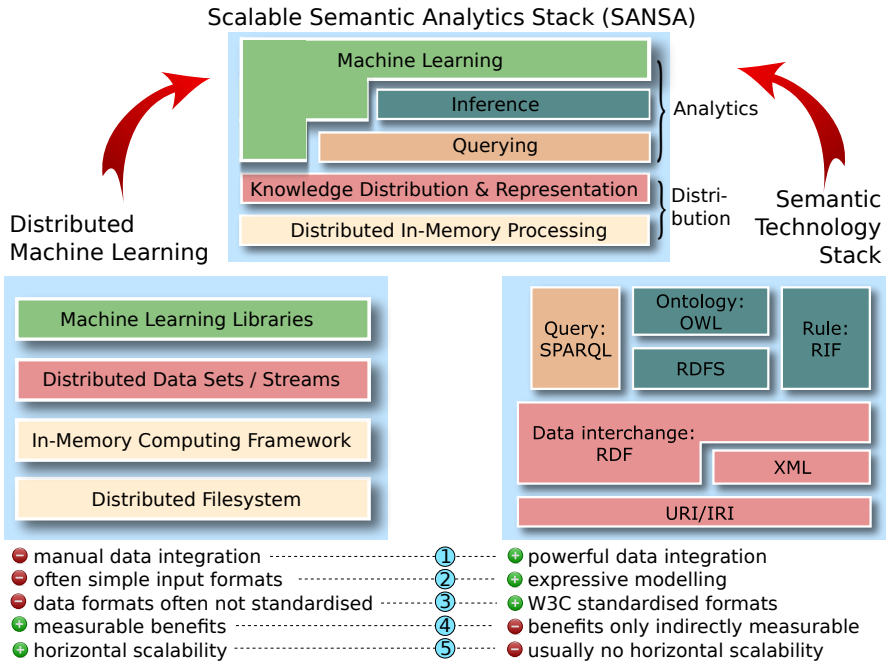

Fig. 1. The SANSA framework combines distributed analytics (left) and semantic technologies (right) into a scalable semantic analytics stack (top). The colours encode what part of the two original stacks influence which part of the SANSA stack. A main vision of SANSA is the belief that the the characteristics of each technology stack (bottom) can be combined and retain the respective advantages.

Query Layer[11] Querying an RDF graph is the primary method for searching, exploring, and extracting information from the underlying RDF data. SPARQL[12] is the W3C standard for querying RDF graphs. Our aim is to have cross-representational transformations and partitioning strategies for efficient query answering. We are investigating the performance of different data structures (e.g., graphs, tables, tensors) in the context of different types of queries and workflows. SANSA provides APIs for performing SPARQL queries directly in Spark and Flink programs. It also features a W3C standard compliant HTTP SPARQL endpoint server component for enabling externally querying the data that has been loaded using its APIs. These queries are eventually transformed into lower-level Spark/Flink programs executed on the Distribution & Representation Layer. At present, SANSA implements flexible triple-based partitioning strategies on top of RDF (such as predicate tables with sub-partitioning by datatypes), which will be complemented with sub-graph based partitioning strategies. Based on the partitioning and the SQL dialects supported by Spark and Flink, SANSA provides an infrastructure for the integration of existing SPARQL-to-SQL rewriting tools. This bears the potential

[11] https://github.com/SANSA-Stack/SANSA-Query
[12] https://www.w3.org/TR/rdf-sparql-query/

advantage of leveraging the optimizers of both the rewriters as well as those of the underlying frameworks for SQL. Currently, the Sparqlify[13] implementation serves as the baseline. Query results can then be further processed by other modules in the SANSA Framework.

Inference Layer[14] Both RDFS and OWL contain schema information in addition to assertions or facts. The core of the forward chaining inference process is to iteratively apply inference rules on existing facts in a knowledge base to infer new facts. This process is helpful for deriving new knowledge and for detecting inconsistencies. Currently, SANSA supports efficient algorithms for the well-known reasoning profiles RDFS (with different subsets) and OWL-Horst, future releases will contain others like OWL-EL, OWL-RL and OWL-LD. In addition, SANSA contains a preliminary version of an adaptive rule engine that can derive an efficient execution plan from a given set of inference rules by generating, analysing and transformation of a rule-dependency graph. By using SANSA, applications will be able to fine tune the rules they require and – in case of scalability problems – adjust them accordingly.

Machine Learning Layer[15] While the majority of machine learning algorithms use feature vectors as input, the machine learning algorithms in SANSA exploit the graph structure and semantics of the background knowledge specified using the RDF and OWL standards. Similar to Markov Logic Networks [16], this enables the algorithms to exploit the expressivity of semantic knowledge structures and potentially attain better performance or more human-understandable results. At the moment, the machine learning layer contains distributed implementations of link prediction algorithms based on two knowledge graph embedding models, namely Bilinear-Diag [24] and TransE [3], and scalable algorithms for RDF data clustering and association rule mining. Effectively and efficiently distributing data structures in potentially complex machine learning approaches is a major challenge in this layer.

3 Use Cases

The main goal of the SANSA framework is to build a generic stack which can work with large amounts of linked data, offering algorithms for scalable, i.e. horizontally distributed, semantic data analysis. To validate this, we are developing use case implementations in several domains and projects.

Life Sciences – Open PHACTS The Open Pharmacological Concepts Triple Store (Open PHACTS)[16] discovery platform provides open access to pharmaceutical data which is gathered and structured through multiple efforts, e.g. Uniprot, GOA, ChEMBL, OPS Chemical Registry, DisGeNET, OPS Identity Mappings, WikiPathways, Drugbank, ConceptWiki and ChEBI, with 2.8 billion triples [18]. Even though this data can potentially fit into the memory of a server (efficient compression techniques in triple stores can compress it to 100 GB), intermediate results of query joins, inference and machine learning algorithms do not fit into memory. For example, our initial experiments

[13] https://github.com/AKSW/Sparqlify
[14] https://github.com/SANSA-Stack/SANSA-Inference
[15] https://github.com/SANSA-Stack/SANSA-ML
[16] https://www.openphacts.org

have shown that even light-weight inference and analysis for a subset of the used data sources (specifically UniProt, EggNOG, StringDB) cannot be efficiently performed on single machines even with 1 TB of main memory. For this reason, distributed approaches are relevant for Open PHACTS. Specifically, they have developed workflows for key questions on the platform [5] which are then used to elaborate API calls that need to be executed. Open PHACTS is currently investigating SANSA as a scalable alternative to perform these workflows over their continuously growing datasets. For example, to answer Question(Q) 6 – "For a specific target family, retrieve all compounds in specific assays" – the task is to look for a particular target family (from the ChEMBL protein classification) and retrieve compounds acting on members of that family (from ChEMBL). SANSA aims to optimise this and similar queries by making use of efficient distributed indexing/querying techniques. SANSA is also under consideration to help in answering complex questions for Open PHACTS, which do not even have a workflow e.g. Q2- "For a given compound, what is its predicted secondary pharmacology?". Tasks like this can be solved by using predictive machine learning models integrated with knowledge graph models, i.e. to search for the primary pharmacology and predicting the associated secondary pharmacologies.

Big Data Platform – BDE Big Data Europe(BDE)[17] [2] is a large Horizon2020 funded EU project which offers an open source big data processing platform allowing users to install numerous big data processing tools and frameworks. The platform is being tested and used by the 17 different partners of the project scattered across Europe and its 7 different use cases cover a variety of societal challenges like climate, health, weather etc. As a specific example, SANSA can be used for log analysis in the context of the BDE platform. The mu.semte.ch micro service in BDE transforms docker events to RDF and stores them in a triple store. Work is also being done in order to translate HTTP network traffic to RDF. The data from these logs (events and HTTP traffic) can be then combined with the data for a particular micro service and its relevant load (CPU/memory usage) on the server. SANSA can then build a predictive cost model for the micro service calls. This can further be extended for efficient resource allocation, monitoring and creation of common user profiles.

Publishing Sector – Elsevier Semantic technologies are very useful in the publishing industry. For example, with in-depth medical knowledge and more than 400 000 scientific articles published per year, annotated with more than 8 million entities and mappings to the Elsevier Merged Medical Taxonomy (EMMeT), Elsevier is building up and testing a large-scale knowledge graph. Elsevier is currently applying (and approaching the limits of) state-of-the-art matrix and tensor factorisation methods, which will be distributed and enhanced in SANSA. There are at least three critical application areas for the methods developed in SANSA: 1.) entity resolution (of author profiles, organisation profiles, etc.), 2.) semantic querying in complex databases (e.g. Clinical Key) and 3.) taxonomy construction. At present publishers, and Elsevier specifically, have to resort to methods which are less accurate than the state of the art due to scalability problems.

Education Sector – University of Bonn While not an external use case, the university labs [18] in which we use SANSA have also further progressed and we now have 12

[17] https://github.com/big-data-europe

[18] http://sda.cs.uni-bonn.de/teaching/sose2017dbda/

students divided into 7 groups using the framework and implementing different scenarios using SANSA functionalities. There are also at least seven students conducting their master thesis on top of the SANSA framework.

Proprietary Data Analytics – Ten Force Ten Force is using SANSA for the clustering of the ESCO,[19] and their proprietary data to analyse the grouping of skills and occupations. Tenforce is also in the process of using association rule mining on their proprietary data to analyse the shopping baskets.

4 Related Work

We give a brief and incomplete account of existing work in distributed RDF querying, inference and machine learning focusing on approaches available as software frameworks.

Querying: SparkRDF [23] and H2RDF+ [15] use RDF dataset statistics to find best merge-join orders for efficient querying. Huang et al. [12] present a hybrid system using in-memory retrieval and map-reduce. TriAD [11] is a specialised shared nothing system that was later [13] improved by using dynamic data exchange for join evaluation. SPARQLGX [9] is an approach for a distributed RDF querying which translates SPARQL to Spark operations. SANSA partially includes the Spark-based S2RDF [17] querying engine which rewrites SPARQL queries to SQL. SANSA facilitates the integration of existing engines under a uniform set of APIs and extends the state of the art in querying through new distributed indexing and partitioning strategies.

Inference: Different distributed rule-based approaches, optimised for one of the many language profiles for the semantic web, have been developed in the past. A scalable distributed reasoning for RDFS entailment rules introduced by Urbani et al. [20], uses optimal execution ordering of the rules to reduce computation time. The WebPIE [19] forward chaining reasoner uses a MapReduce approach. QueryPie [21], uses backward chaining and distributes the schema triples. Cichlid [10] is a distributed reasoning engine, using the Apache Spark framework. The above systems only support (fragments of) the OWL RL language profile. SANSA provides a general rule-based reasoning engine that optimises executions plans for an arbitrary set of rules by taking into account the logical dependencies between rules, the distribution of the data w.r.t. the rules, and the technical features of the underlying distributed processing framework.

Machine Learning: There are numerous centralised machine learning frameworks and algorithms for RDF data. DL Learner [4] is a framework for inductive learning for the Semantic Web. AMIE [8] learns association rules from RDF data. ProPPR [22] and TensorLog [6] are recent frameworks for efficient probabilistic inference in first order logic. Nickel et al. provide a review of statistical relational learning techniques for knowledge graphs [14]. Scaling up structured machine learning algorithms, which are mostly iterative convergent in nature, using Bulk Synchronous Parallel frameworks (e.g. Spark, Flink) is a challenging task.

General: Previous approaches demonstrate specialised efforts related to specific layers of the SANSA stack. In contrast to this, SANSA provides a unified platform for distributed machine learning over large-scale knowledge graphs, combined with querying and rule-based inference. This makes it easier for developers to access its functionality,

[19] `https://ec.europa.eu/esco/portal/escopedia`

move between different implementations and assemble existing functionality into larger workflows. To the best of our knowledge, SANSA is the only holistic framework for distributed analytics on large-scale RDF data.

5 Conclusions and Future Work

We presented the SANSA framework, which combines the advantages of distributed in-memory computing and semantic technologies. Its holistic layered approach leverages data integration and modelling capabilities provided by semantic technologies with machine learning functionality and improved horizontal scalability provided by distributed in-memory frameworks. We believe that SANSA is an important framework for the semantic technology community as well as those parts of the distributed in-memory development community which require more sophisticated data modelling capabilities. In the future, we will enrich SANSA with algorithms for inference-aware knowledge graph embeddings, distributed approximate reasoning and further data partitioning strategies.

Acknowledgements This work was partly supported by the grant from the European Union's Horizon 2020 research Europe flag and innovation programme for the project Big Data Europe (GA no. 644564) and a research grant from the German Ministry BMWI under the SAKE project (Grant No. 01MD15006E).

References

1. J. S. Andersen and O. Zukunft. Evaluating the Scaling of Graph-Algorithms for Big Data Using GraphX. In *International Conference on Open and Big Data (OBD)*, pages 1–8. IEEE, 2016.
2. S. Auer et al. The BigDataEurope Platform - Supporting the Variety Dimension of Big Data. In *17th International Conference on Web Engineering (ICWE2017)*, 2017.
3. A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.
4. L. Bühmann, J. Lehmann, and P. Westphal. DL-Learner—A framework for inductive learning on the Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 39:15–24, 2016.
5. C. Chichester, D. Digles, R. Siebes, A. Loizou, P. Groth, and L. Harland. Drug discovery FAQs: workflows for answering multidomain drug discovery questions. *Drug discovery today*, 20(4):399–405, 2015.
6. W. W. Cohen. TensorLog: A differentiable deductive database. *arXiv preprint arXiv:1605.06523*, 2016.
7. I. Ermilov, J. Lehmann, M. Martin, and S. Auer. LODStats: The data web census dataset. In *International Semantic Web Conference*, pages 38–46. Springer, 2016.
8. L. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek. Fast Rule Mining in Ontological Knowledge Bases with AMIE+. *Very Large Databases Journal*, 2015.
9. D. Graux, L. Jachiet, P. Genevès, and N. Layaïda. *SPARQLGX: Efficient Distributed Evaluation of SPARQL with Apache Spark*, pages 80–87. Springer International Publishing, Cham, 2016.

10. R. Gu, S. Wang, F. Wang, C. Yuan, and Y. Huang. Cichlid: efficient large scale RDFS/OWL reasoning with spark. In *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*, pages 700–709. IEEE, 2015.
11. S. Gurajada, S. Seufert, I. Miliaraki, and M. Theobald. TriAD: A Distributed Shared-nothing RDF Engine Based on Asynchronous Message Passing. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 289–300, New York, NY, USA, 2014. ACM.
12. J. Huang, D. J. Abadi, and K. Ren. Scalable SPARQL Querying of Large RDF Graphs. *PVLDB*, 4(11):1123–1134, 2011.
13. Y. Nenov, R. Piro, B. Motik, I. Horrocks, Z. Wu, and J. Banerjee. RDFox: A highly-scalable RDF store. In *International Semantic Web Conference*, pages 3–20. Springer International Publishing, 2015.
14. M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.
15. N. Papailiou, I. Konstantinou, D. Tsoumakos, and N. Koziris. H2RDF: adaptive query processing on RDF data in the cloud. In *Proceedings of the 21st International Conference on World Wide Web*, pages 397–400. ACM, 2012.
16. M. Richardson and P. Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.
17. A. Schuetzle, M. Przyjaciel-Zablocki, S. Skilevic, and G. Lausen. S2RDF: RDF Querying with SPARQL on Spark. *PVLDB*, 9(10):804–815, 2016.
18. A. Troumpoukis, A. Charalambidis, G. Mouchakis, S. Konstantopoulos, R. Siebes, V. de Boer, S. Soiland-Reyes, and D. Digles. Developing a Benchmark Suite for Semantic Web Data from Existing Workflows. In *BLINK@ISWC*, 2016.
19. J. Urbani, S. Kotoulas, J. Maassen, F. Van Harmelen, and H. Bal. OWL reasoning with WebPIE: calculating the closure of 100 billion triples. In *Extended Semantic Web Conference*, pages 213–227. Springer Berlin Heidelberg, 2010.
20. J. Urbani, S. Kotoulas, E. Oren, and F. Van Harmelen. Scalable Distributed Reasoning Using MapReduce. In *International Semantic Web Conference*, pages 634–649. Springer Berlin Heidelberg, 2009.
21. J. Urbani, F. Van Harmelen, S. Schlobach, and H. Bal. QueryPIE: Backward reasoning for OWL Horst over very large knowledge bases. In *International Semantic Web Conference*, pages 730–745. Springer Berlin Heidelberg, 2011.
22. W. Y. Wang, K. Mazaitis, and W. W. Cohen. Structure learning via parameter learning. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1199–1208. ACM, 2014.
23. Z. Xu, W. Chen, L. Gai, and T. Wang. SparkRDF: In-memory distributed RDF management framework for large-scale social data. In *International Conference on Web-Age Information Management*, pages 337–349. Springer, 2015.
24. B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.