

Towards a Semantic Clinical Data Warehouse: A Case Study of Discovering Similar Genes

Benedikt Kämpgen¹, Horst Werner², Radwan Deeb², and Christof Bornhövd²

¹ FZI Research Center for Information Technology, Karlsruhe, Germany,
kaempgen@fzi.de

² SAP AG, Karlsruhe, Germany,
firstname.lastname@sap.com

Abstract. Physicians nowadays have to consider a diverse range of data sources when treating a patient. Semantic clinical data warehouses allow to easily add new data and to pro-actively help the physician making sense of the data. In this work-in-progress paper we investigate an approach of using Linked Data as the access mechanism and a graph database for storage and query processing. We describe lessons learned from a case study of discovering similar genes where we use an existing similarity metric to derive new information, the Gene Ontology as a data source, and SAP HANA as an efficient graph database.

1 Introduction

Examples of data sources that physicians nowadays have to consider when treating a patient include background information collected as part of trials or from publications and encyclopedias, as well as genomic information [5, 4].

An example tool to support the physician in accessing and analysing the data from such sources is the Patient Data Explorer (PDE) based on the SAP HANA in-memory database deployed at the National Center for Tumor Diseases (NCT) in Heidelberg³. PDE allows an overview of patients; to create diagrams visualising the distribution of characteristics in patients; and to zoom-in to single patients to get detailed information about diagnoses and therapies.

PDE can be improved in several ways: PDE uses a broadly-applicable entity relationship data model about “interactions” and “observations” similar to a Star Schema; adding additional background information such as ICD codes or PubMed references would require to manually modify ETL pipelines and the schema. For information coming from different sources heterogeneities remain, e.g., different terminologies for diseases and drugs may be used and inconsistencies and redundancies easily occur. Maschine Learning (ML) algorithms such as for clustering of genes are difficult to apply for physicians and the results are not written back to the data warehouse for provenance tracking and information sharing.

³ <http://www.sap-innovationcenter.com/2013/09/19/medical-research-insights/>

In this work-in-progress paper we argue that overcoming such challenges is possible using Linked Data, graph databases, and semantic algorithms (Section 2): we describe a use case for discovering similar genes (Section 3) and derive lessons learned (Section 4). We mention related work (Section 5) and conclude (Section 6).

2 Semantic Clinical Data Warehouse

See Figure 1 for the architecture. Information in the semantic clinical data warehouse is presented to the user by a visualisation and analysis tool. To store, query, and visualise arbitrary information we use a graph database and the following intuitive data model (property graph): Relevant objects such as patients, interactions, and observations are represented as vertices in the graph. Such objects have properties with values of primitive datatypes such as String and Integer, e.g., the surname of a patient. Objects are related to each other via edges in the graph, e.g., a patient is diagnosed with a disease. Such relationships also can have properties, e.g., provenance information about the algorithm or human expert that has generated the relationship. The integrator and reasoner component 1) translates an RDF graph to a property graph, 2) derives implicit information useful for data integration and decision support of users, and 3) imports the graph to the graph database. The RDF graph is crawled based on the Linked Data principles.

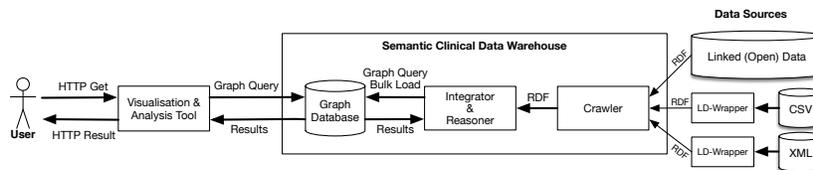


Fig. 1. Semantic clinical data warehouse based on Linked Data and graph database

This architecture has the following advantages: Already, there are large amounts of life science data – directly or using LD wrappers – published using such widely-adopted access mechanisms and standard vocabularies [1]. A graph database is more schema-flexible than a relational database, i.e., if new data sources introduce new vertices, edges, and properties in the graph, no database administrator has to modify the schema. Linked Data allows to easily add new data sources to the data warehouse by following new links to further objects on the Web. Implicit information can be derived in two ways: 1) by evaluating OWL axioms represented in RDF; for instance, semantics from the OWL 2 RL profile such as equality can be evaluated using rule engines, and 2) by ML algorithms that make use of ontological information, e.g., to discover similar genes. Also, graph databases usually are designed to efficiently process analytical operations over large graphs, i.e., can be used to efficiently compute and write-back results from ML algorithms.

3 Case Study of Discovering Similar Genes

In this section, we apply our approach to a use case for discovering similar genes from a plant [4]. Similarity is an important basis for other relationships. For instance, the effect of a drug depends on the genes it targets. If drugs target similar genes, they likely have similar effects.

Relevant data sources for our prototype – HANA Linked Data AnnSim (HLA) – are descriptions of genes⁴, gene annotations from experts⁵, and the Gene Ontology (GO) with a concept hierarchy⁶.

Using *OpenRefine with RDF extension*, we translate the former two sources to RDF and reuse links from the GO RDF representation. Crawling such data results in one RDF graph with genes, concepts, and annotations between genes and concepts.

HLA uses as a graph database *HANA Graph*, an extension to the HANA in-memory database for storing and querying of property graphs [6]. Graphs are logically stored in HANA using two (virtual) tables: one table for vertices and one table for edges each with columns for an id and every possible property. Graph queries over HANA Graph are issued using the so-called *GEM* language and are translated to SQL queries over the two tables. Based on a column-oriented and in-memory database, HANA Graph allows fast query processing.

An importer program then maps the crawled RDF graph to a property graph and bulk loads the property graph to HANA Graph. Intuitively, the importer generates for every triple two vertices for the subject and object (if not existing), and an edge for the predicate. HANA Graph then contains genes (e.g., AT5G23810) and concepts (e.g., Amino Acid Transport) as vertices, and relationships between genes and concepts as edges. For instance, there are annotation relationships between genes and concepts as well as is-a relationships between concepts. Vertices and edges can have properties, e.g., a concept has a textual description. The graph is then extended with edges between genes describing their similarity, and edges between concepts describing their distance in the is-a concept hierarchy.

Such information we compute based on an existing algorithm, *AnnSim* [4]. AnnSim makes use of the distances between the concepts of two genes. Intuitively, the shorter the average path between any two concepts of two genes the more similar the two genes. Both computed distances and similarities are written back to HANA Graph as edges between concepts and genes, respectively. Every edge has as a property a numeric value between 0 and 1 for the similarity and distance, and – in case several different algorithms are used – the name of the algorithm.

⁴ ftp://ftp.arabidopsis.org/home/tair/Genes/TAIR10_genome_release/TAIR10_functional_descriptions

⁵ ftp://ftp.arabidopsis.org/home/tair/Ontologies/Gene_Ontology/ATH_GO_GOSLIM.txt

⁶ <http://purl.obolibrary.org/obo/go.owl>

As also visible in a screencast on our paper website⁷, the user of HLA gets an overview of genes and their similarities to other genes; can zoom into single genes to see textual descriptions of concepts; can visit concepts along the concept hierarchy (Figure 2). Also, the user can ask for a graph view showing similarities between genes based on distances between concepts (Figure 3). For the visualisation, we used a visualisation engine called *Symbiosis* that can be configured with a JSON-based template language to visualise a graph. Symbiosis uses HANA Graph and GEM for querying the data.



Fig. 2. Zoom-in to single gene (left) and **Fig. 3.** Graph view of genes with smaller distances displayed concept hierarchy (right)

4 Lessons Learned

To draw preliminary lessons learned about the applicability of our approach, we compared our HLA system with an implementation of AnnSim by Palma et al. (AnnSim 1.0) [4] to compute pair-wise similarities of 20 genes from the taxonomic class *1-aaap*.

For both approaches, we use a workstation with Ubuntu 14.04 VM on W7 Intel Core i5-3360M CPU 2.80GHz, 16 GB RAM to execute the program logic. In addition to that, for HLA, we use a server with SUSE Linux Enterprise Server 11.1 500 GB RAM, 80 cores to host HANA Graph. Table 1 compares the two approaches. HLA uses the same data sources than AnnSim 1.0 but considers all contained information and the newest versions.

Table 1. Available relevant data for HLA and AnnSim 1.0

Approach	Size of data	Triples	Vertices	Edges
HLA	537 MB	7,337,447	601,519	1,658,322
AnnSim 1.0	2.80 MB	-	39,209	74,123

Correct Computation of Similarities. There is a mean squared error between the results of HLA and AnnSim of 0.09. This difference we expect is due to newer, possibly more elaborate versions of annotations and GO (version 1.2) used by HLA. We compared the results of both approaches with the gold standard, a similarity metric based on the DNA sequence of genes (SeqSim).

⁷ <http://people.aifb.kit.edu/bka/hla/>

The mean squared error between HLA and SeqSim (0.19) is lower than between AnnSim 1.0 and SeqSim (0.36), indicating that AnnSim similarities improve with newer data sources; yet, further experiments are needed to confirm this claim.

Scalable Computation of Similarities. Table 2 gives an overview of the time for the different steps in the execution. Loading of data is estimated with a connection of 6.7 Mbps download speed. Although HLA takes considerably longer than AnnSim 1.0, we argue that HLA’s bottlenecks can be resolved and that HLA is more promising for larger datasets.

Table 2. Elapsed query processing time (in sec) for computing similarities of 20 genes

Approach	Prepare Sources	Download Data	Map Graph	Load Graph	Compute AnnSim	Read Queries	Write Queries
HLA	< 120	641	355	15	2,667	230	2,202
AnnSim 1.0	N/A	3	0	0	408	-	-

AnnSim 1.0 uses a proprietary graph data format with reduced information that is probably fast to generate (Prepare), download, and load. HLA uses a more verbose but also more expressive graph model (RDF) and has to generate (Prepare), download, transform to property graph (Map) and load 15 times more vertices, 22 times more edges and comprehensive properties such as textual descriptions. Loading graph data to HANA Graph showed fast and the preprocessing steps we believe can be optimised by parallelisation.

AnnSim 1.0 uses program logic in C/C++ over arrays to compute the 400 similarities and displays the results to users. HLA loads the relevant data to a graph database and uses program logic in Java to issue database queries to efficiently compute the similarities and to write back the results to the data warehouse. The query language GEM was useful and intuitive for graph-traversal queries. For instance, the following GEM read query is issued using a special-type function WIPE() to the SQL interface of HANA, recursively visiting one or more edges of type `rdfs:subClassOf`, and returns a vertex table with all ancestors of a GO concept: `RESULT uri:myResult FROM { GO:0005634 }-[@core:type = 'rdfs:subClassOf']->(1,*)`;

The program logic in HLA spent more than 90% of the time to compute a specific part of AnnSim, a min-weight perfect matching (Blossom IV). We believe we can optimise the Blossom IV execution, e.g., by running part of it directly in HANA Graph via built-in and user-defined functions. Writing back of the results to the data warehouse took a lot of time since done using single write queries instead of a bulk load. In this case, since read queries to HANA Graph showed fast, HLA should also scale with larger datasets, in contrast to AnnSim 1.0 that does not outsource bulk loading, reading, and writing to an external database. Computing additional information can be done offline. Interactive visualisation over HANA Graph were possible using the Symbiosis engine.

Flexible Computation and Visualisation of Similarities. Whereas AnnSim 1.0 was implemented specifically for the problem of efficiently computing similarities of objects described in a proprietary format, HLA uses Linked Data as a unified data model and standard access mechanism.

New data sources can be added to HLA by providing more links to crawlable Linked Data. We believe that efforts such as by Bio2RDF [1] to release life science Linked Data will allow to semi-automatically resolve semantic conflicts using OWL semantics and rules.

Other objects such as patients can be compared in HLA; AnnSim only requires objects to be annotated with concepts and concepts to be described in an is-a hierarchy. Algorithms that use other relationships and derive other information can be added to HLA. The Symbiosis engine showed that – given sufficient understanding of the domain experts’ problem – it is easily possible (5–10h of manual work) to provide flexible visualisations over a graph-based data model.

5 Related Work

According to Haussler et al. [3] a Million Genome Warehouse has to pro-actively process relevant data in data analysis pipelines to draw valid and useful medical inferences. HLA accesses the Gene Ontology and computes AnnSim [4], yet, can be extended with other biomedical ontologies and other semantic similarity measures [5]. HLA uses the HANA Graph in-memory database [6] but may also use other graph databases such as Graphium. Callahan and Dumontier [2] present an approach to represent and evaluate scientific hypotheses based on RDF and SPARQL.

6 Conclusions

In this work-in-progress paper, in a small case study of discovering similar genes we illustrated the potential of modular access mechanisms with Linked Data, queries over a schema-flexible graph database, and semantic algorithms to derive new information. Continuously adding new data sources and data items, new algorithms, and new visualisations leave exciting future work.

References

1. Callahan, A., Cruz-Toledo, J., Ansell, P., Dumontier, M.: Bio2RDF Release 2: Improved Coverage, Interoperability and Provenance of Life Science Linked Data. In: *The Semantic Web: Semantics and Big Data* (2013)
2. Callahan, A., Dumontier, M.: Evaluating Scientific Hypotheses Using the SPARQL Inferencing Notation. In: *The Semantic Web: Research and Applications* (2012)
3. Haussler, D., Patterson, D., Diekhans, M., Fox, A., Jordan, M., Joseph, A., Ma, S., Paten, B., Shenker, S., Sittler, T., Stoika, I.: A Million Cancer Genome Warehouse. Tech. rep., University of California at Berkeley (2012)
4. Palma, G., Vidal, M.E., Haag, E., Raschid, L., Thor, A.: Measuring Relatedness Between Scientific Entities in Annotation Datasets. In: *International Conference on Bioinformatics, Computational Biology and Biomedical Informatics* (2013)
5. Pesquita, C., Faria, D., Falco, A.O., Lord, P., Couto, F.M.: Semantic Similarity in Biomedical Ontologies. *PLOS Computational Biology* 5(7) (2009)
6. Vasilyeva, E., Thiele, M., Bornhövd, C., Lehner, W.: Leveraging Flexible Data Management with Graph Databases. *First International Workshop on Graph Data Management Experiences and Systems* (2013)