Crumbs4Cube: Turning Breadcrumbs into Smart Enriched Data Cubes

Dihia Lanasri[1], Selma Khouri[1], Roaya Saidoune[1], Kamila Boudoukha[1], and Ladjel Bellatreche[2]

[1] ESI, Algiers, Algeria
[2] LIAS/ISAE-ENSMA, Poitiers, France

Abstract. Traditionally, companies analyze data that has been extracted from their *internal* data sources and loaded into data warehouses (DWs). In the digitalization Era, the data that companies are looking for is no longer limited to their internal sources, but also include external resources such as Linked Open Data (LOD). Usually, these sources are composed of two components: datasets and data left behind by people while exploring these data (digital breadcrumbs materialized by *SPARQL query logs*). Considering the breadcrumbs in the process of DW construction represents a big challenge, due to their volume, variety and the expertise of users generating them. In this paper, we propose *Crumbs4Cube*, a comprehensive tool for discovering multidimensional patterns from LOD breadcrumbs. It offers designers mechanisms to manage, investigate and visualize these breadcrumbs before integrating them into a target DW.

Keywords: Multidimensional patterns, LOD query-logs, LOD datasets, Smart Data Discovery

1 Introduction

Companies never stop integrating data from their local sources into their data warehouses (DWs) to get added value. Local sources are usually insufficient to let these companies being more competitive in making intelligent decisions. In this context, they are obliged to include smart external resources to enrich their internal sources to get advanced insights. Under the impulse of the open-world data, the enrichment technique has shown its benefit in terms of the added value of designing several constantly evolving digital entities such as *knowledge graphs* [2] and *data repositories* [9]. This enrichment is usually performed by resorting to external sources such as sensors, social network messages, and LOD. These sources are usually composed of two components: **(i)** datasets and **(ii)** the data left behind by people while exploring these datasets. LOD is one of the valuable external sources. Their principles allow naive and expert people accessing and retrieving data stored and published via SPARQL endpoints reflecting their requirements. Various initiatives like USEWOD and LSQ collected

and published these query-logs that may contain valuable multidimensional patterns (MDP) that include facts, dimensions, and hierarchies. Face to this situation, these crumbs have to be investigated to *turn their hidden data into smart multidimensional data that enriches an existing DW*.

In the last decade, a couple of studies proposed a materialized integration of the first component of the LOD (i.e. the datasets) in the DW design [4,5], where multidimensional models are derived from the LOD datasets. In these proposals, the LOD datasets undergo the same DW processes performed for internal sources. Other studies analyzed the LOD dataset via standard OLAP operations using tools such as SPARQLytics [7]. These studies ignore the second component of the LOD: the query-logs. Integrating these crumbs in the DW construction is not an easy task, because they are large, contain hidden smart multidimensional data, do not have the same structure and format (graphs) as the internal sources. To facilitate this integration, these crumbs have to be prepared in terms of cleaning, exploration, investigation [3] and visualization in order to extract smart multidimensional data. Once prepared, they will be integrated into the DW. In this study we focus on the investigation of the query-logs for discovering MDP [1], the enrichment process of the MDP in an existing DW is not supported by our tool, it requires schema matching and integration techniques that have been widely discussed in literature.

In this paper, we propose a tool *Crumbs4Cube* for augmenting an existing DW through automatic discovering of MDP from the LOD query logs and creating their associated conceptual multidimensional graph that can be exploited by designers to investigate the hidden data in the query logs. This paper is organized as follows: Section 2 presents the architecture of Crumbs4Cube. Section 3 presents our demonstration and Section 4 concludes our paper.

2 System Architecture Overview

Crumbs4Cube generates a conceptual multidimensional graph based on the Fact/dimension dichotomy. Our application adopts a 3-tiered architecture based on the MVC framework.

I) Data layer: in this data layer, Jena TDB triple store is used to persist the RDF graphs. LOD are defined using RDF standard presenting data in the form of RDF graphs. Each RDF statement is a directed labeled graph and takes the form <s, p, o, g> such that in the graph label g, subject s has the predicate (i.e. property) p, and the value of that property is the object o. Sparql queries are defined for matching a defined subgraph of triples $< s', p', o' >$ in the queried RDF graph. For instance, *Example_1* illustrates a Sparql query of scholarly data LOD used in our demonstration:

```
SELECT DISTINCT ?pred ?author_url ?author_name
WHERE { <uri/conquer-query> bibo:authorList ?authorList. ?authorList ?pred ?author_url .
?author_url foaf:name ?author_name}
```

II) Business layer: the different modules of *Crumbs4Cube* (Fig. 1) are developed using JAVA and the Jena API (ARQ and Core libraries) as follows:

(1) Clean log-queries. It consists in syntactic and semantic cleaning of SPARQL queries in order to keep valid SELECT queries, using REGEX, HTML-parser (to decode queries) and ARQ JENA (SPARQL parser).

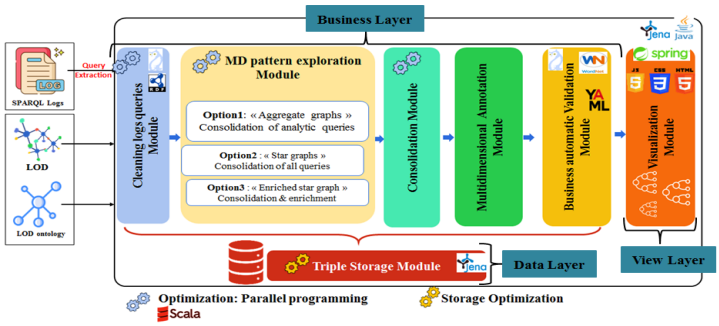

Fig. 1. Crumbs4Cube architecture

(2) MDP Exploration. It is the core of our process. It consists in constructing the star graphs from the cleaned queries using three identified scenarios:

Scenario 1: Aggregate Queries. It starts by identifying the aggregate queries among the set of valid queries. In such queries, the measures of facts are already aggregated via functions like: COUNT, SUM, etc. and analyzed by some dimensions identified in the GROUP BY clause, the identification of facts and dimensions is consequently straightforward. Aggregate queries are of the form:

```
Select ?var1 ?var2 ... ?varn  AggFct1(?varx1) ... AggFct(?varxm)}
Where {triple patterns <S,P,O>} Group By ?var1,..., ?varn
```

Scenario 2: Star Graphs. Unlike the first scenario, this scenario investigates the interactions between queries. The MDP [6] are identified from the consolidation of the different graphs generated from the all valid queries.

Scenario 3: Enriched Star Graphs. It extends scenario 2 by enriching the obtained MDP with new concepts identified from the LOD dataset.

For **Scenario 1**, the following steps are performed for each query: *(a) Aggregate query construction:* for identifying the variables in each triple <?S,P,?O> of the query (like *?pred* or *?author_url* in the query of *Example_1* given above), we defined BPEnriched function that analyses the queries by: (i) adding a triple $<?S$,*rdf:type*,*?type* $>$ to retrieve ?type reflecting the concept of $?S$ (same process is achieved for variables ?O), (ii) adding a triple $<?P$,*rdf:range*,*?type* $>$ for identifying the *Datatype* of property P, *(b) Query execution:* executes the queries enriched by the cited triples on the SPARQL endpoint and retrieves the results. *(c) Graph construction:* from the obtained results, the process constructs the multidimensional graph of each query, where: the attributes on which aggregate functions are applied are considered as measures. The concepts of these attributes are considered as facts, the concepts associated to the Group By are considered as dimensions or dimension attributes according to their type. These MDP are constructed as graphs where the vertex is the fact and dimensions & attributes are the related nodes.

For **scenarios 2 and 3**, the following steps are performed: *(a) Transform Select to Construct Query:* after enriching the queries using BPEnriched function, one main step that is specific to scenarios 2 and 3 is the construction of a graph reflecting the triple patterns of each query, using a Construct query. *(b)*

Query Execution: it executes the Construct queries on the SPARQL endpoint then retrieves the results as an RDF graph. *(c) Graph alleviation*: It consists in lightening graphs by removing triples containing generic properties including: rdf:type, rdf:label, etc. and generic classes like owl:Thing that are not relevant for MDP. *(d) Consolidation:* it consists of two main phases: *(i) Consolidation by vertex*: the obtained graphs sharing the same vertex are consolidated. *(ii) Consolidation by nodes*: for each non-vertex node in a given graph, if it exists as a vertex in another graph G, the children of this node are consolidated with this vertex of G. All the following steps are applied for all scenarios.

(4) Graphs alleviation. A valid multidimensional graph has at least a fact with a dimension and a measure. Non valid graphs are rejected.

(5) Multidimensional Annotation. It allows annotating the multidimensional graphs by adding triples of type <Class, annotated, Annotation>, such as: Class represents the node of the graph, and Annotation ∈ { Fact, Dimension, Leveldimension, Factattribute, Dimension Attributes, Measure }.

(6) Graph Enrichment. Following scenario 3, this module looks deeper into the interactions between the query logs and the LOD dataset. The process starts exploring new properties and concepts that enrich the obtained multidimensional graph annotated. For each fact/dimension/level (S) of the consolidated graph, we look for its related triples $< S, P, O >$ in the LOD knowledge base. For each resource (O): if P is datatype property, O is considered as an attribute. If P is a relationship, O is considered as a dimension or level).

(7) Automatic Business Validation. It allows an automatic validation using a domain ontology if available (or at least Wordnet Ontology) using WS4J library. Automatic validation rules are checked defined in [8].

III) View layer: represents the interfaces of *Crumbs4Cube* that: (a) help the designer discovering MDP from LOD. (b) calculate quality metrics identified (saved in YAML files) of the multidimensional model obtained. This layer is developed in HTML/CSS/JavaScript(JS), graphs are represented with JS amCharts4 library. Scala language is used for optimization issues, Apache Maven and Github for versioning management. The source code of our tool *Crumbs4Cube* is available at `https://github.com/SaidouneRouaya/cubeQE`.

3 Demonstration Overview

For the demonstration of *Crumbs4Cube*, we simulate the process of exploring LOD query logs using ScholarlyData logs; for business validation we used the ScholarlyData Ontology. Log files contain 5 485 082 queries. This demonstration was performed on a machine OS Windows 10, 128 GB of RAM, Processor Intel(R) Xeon(R) CPU E5-2673 V4 @ 2.30GHz.

Scenario 1: a small rate (1,54%) of aggregate queries have been obtained from the set of valid queries, which impacted the rate of MDP discovered. This encouraged us to consider the second scenario.

Scenario 2: The resulting rate of valid queries in the logs is about 63,24%. *Crumbs4Cube* allows the designer to launch the process of generating a multidi-

mensional graph from the treated queries. For this scenario and after validation, 26 multidimensional graphs have been discovered from the logs.
Scenario 3: After enriching the multidimensional graph with ScholarlyData dataset, we obtained new results that show new patterns discovered from the dataset (Fig.3 for the concept Workshop_Events). The tool also provides different metrics for all scenarios, used to evaluate the quality of the discovered MDP, they are used to help the BI designer to validate the obtained MD graphs. A demonstration video is available at `https://youtu.be/4Z1X3X12gB4`.

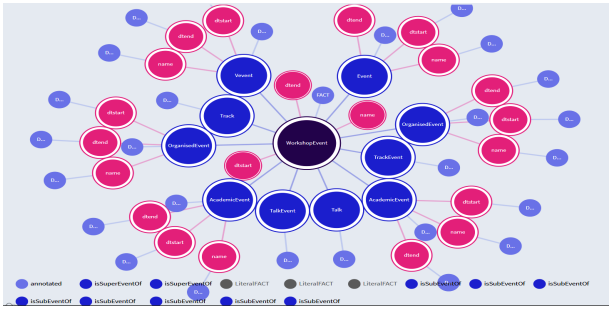

Fig. 2. WorkshopEvents Multidimensional Graph

4 Conclusion

In this paper, we presented a new tool *Crumbs4Cube* that discovers MDP from LOD query-logs exploration. *Crumbs4Cube* is developed to assist the designer through different modules and allows extracting useful multidimensional insights that are difficult to explore directly from the dataset without prior knowledge.

References

1. Khouri, S., Lanasri, D., Saidoune, R., Boudoukha, K., Bellatreche, L.: Loglinc: Log queries of linked open data investigator for cube design. DEXA (2019)
2. Lehmann, J., et al.: Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2), 167–195 (2015)
3. Morton, K., Balazinska, M., Grossman, D., Mackinlay, J.: Support the data enthusiast: Challenges for next-generation data-analysis systems. Proceedings of the VLDB Endowment **7**(6), 453–456 (2014)
4. Nebot, V., Berlanga, R.: Statistically-driven generation of multidimensional analytical schemas from linked data. Knowledge-Based Systems **110**, 15–29 (2016)
5. Rizzi, S., Gallinucci, E., Abelló, M.G.A., Romero, O.: Towards exploratory olap on linked data. In: SEBD. pp. 86–93 (2016)
6. Romero, O., Abelló, A.: Automatic validation of requirements to support multidimensional design. Data & Knowledge Engineering **69**(9), 917–942 (2010)
7. Rudolf, M., H.Voigt, Lehner, W.: Sparqlytics: multidimensional analytics for rdf. Datenbanksysteme für Business, Technologie und Web (BTW 2017) (2017)
8. Salem, A., Ben-Abdallah, H.: The design of valid multidimensional star schemas assisted by repair solutions. VJCS **2**(3), 169–179 (2015)
9. Wang, X., Carey, M.J.: An IDEA: an ingestion framework for data enrichment in asterixdb. CoRR **abs/1902.08271** (2019)