Knowledge hypergraph-based multidimensional analysis for natural language queries: application to medical data

Sana Ben Abdallah Ben Lamine[1] [0000−0002−6018−2518], Marouane Radaoui[1] [0000−0001−8575−3711], and Hajer Baazaoui Zghal[2] [0000−0002−2151−7397]

[1] Riadi Laboratory, ENSI, University of Manouba, Tunisia
sana.benabdallah@riadi.rnu.tn, raddaouimarouen@gmail.com
[2] ETIS UMR8051, ENSEA, CY University, CNRS, France
hajer.baazaoui@ensea.fr

Abstract. In recent years, data is continuously evolving not only in volume but also in types and sources, which makes the multidimensional analysis and decision making using traditional approaches a complex and difficult task. In this paper, we propose a three-layer-based architecture to perform multidimensional analysis of natural language queries on health data: 1/ Treatment layer aiming at xR2RML mappings generation and knowledge hypergraph building; 2/ Storage layer allowing mainly to store the RDF triples returned by the query of NoSQL databases, and 3/ Semantic layer, based on a domain ontology which constitutes the knowledge base for the generation of the mappings and the building of the knowledge hypergraph. The originality of our proposal lies in the knowledge hypergraph and its capacity to support multidimensional queries. A prototype is developed and the experiments have shown the relevance of the returned multidimensional query results as well as an improvement over traditional approaches.

Keywords: Knowledge hypergraph · Multidimensional analysis · Natural language queries · NoSQL databases · Health decision support.

1 Introduction

One of the main needs of decision-makers is going through Data Warehouses (*DW*s), building OnLine Analytical Processing (*OLAP*) cubes and performing MultiDimensional Analysis (*MDA*)[1]. The transition to *NoSQL* systems in *DWs* gives decision-makers the possibility of storing and querying unstructured data [2] in large amounts. *DWs* allow *MDA* but remain an expensive solution. Thus, some works proposed to achieve *MDA* on *NoSQL DBs* without going through *DWs* [4,5]. Nonetheless, for decision-makers, and health experts it seems difficult to formulate a MultiDimensional Query (*MDQ*).

In this paper, we propose an Knowledge HyperGraph(*KHG*)-based approach to perform *MDA* of natural language queries over multi-source health *NoSQL* data aiming at improving the decision-making process. A prototype is developed

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

and the experiments have shown the results' relevance. In the remainder of this paper, a brief overview of related works is given in section 2. Then, sections 3 and 4 detail our proposal and experimental results. Finally, we conclude and present our future works in section 5.

2 Related works

To address *MDA*, *DWs* use *OLAP* to query data and analyze it from multiple perspectives. Nonetheless, relational models, usually used to implement *DWs* don't permit managing massive data, in addition to data non-freshness and cost of *DWs*. To overcome these issues, researchers have proposed alternative solutions for *MDA*. The use of *NoSQL DWs* is increasingly envisaged [2]. In [4,5] *MDA* is done via direct access to a document-oriented *NoSQL DBs*.

On the other hand, Knowledge Graphs (*KGs*) and *KHGs* allow solving interoperability problems in order to interrogate efficiently massive and heterogeneous data [6]. *KGs* have been also attractive to researchers for they support analytics and decision-making [7]. Some works have used *KGs* to address *MDA*. In [11], *OLAP* is adapted to perform analysis on *KGs*. In [12], a graph-based *DW* is proposed. Our motivation is to exploit the advantages of *KHGs* in *MDA*.

3 Proposed approach

Fig. 1 presents the three-layers architecture of our *MDA* approach.

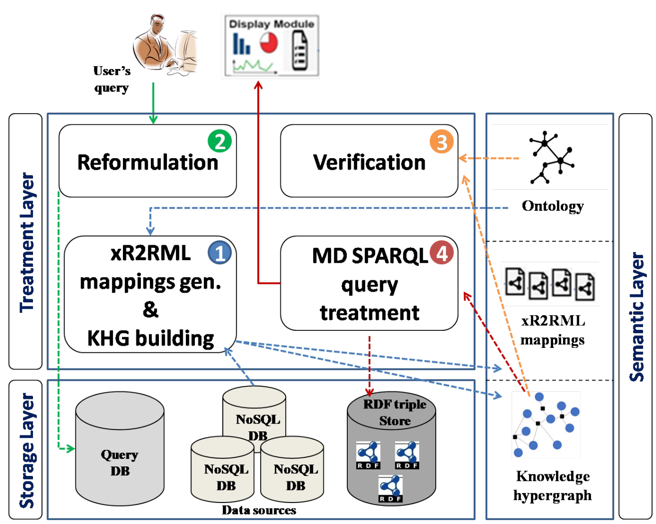

Fig. 1: Knowledge hypergraph-based multidimensional analysis architecture

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

1. **Treatment layer**: its four modules are detailed in the following subsections.
2. **Storage layer**: it consists of:
   – ***NoSQL DBs***: are the data sources targeted by the user's query
   – **Query *DB***: stores the queries and their respective reformulations.
   – ***RDF* triples store**: stores the *RDF* triples returned by the query.
3. **Semantic layer**: it consists of:
   – **Ontology**: it is the domain ontology developed in [9] and constituting the knowledge base for the generation of the mappings and the building of the *KHG*. It is also used in the verification module.
   – **KHG**: is a data integration framework allowing for unified querying.
   – ***xR2RML* mappings**: are *RDF* documents representing logical sources extracted from the input databases.

3.1 *xR2RML* mappings generation and *KHG* building module

***xR2RML* Mappings Generation** Is done in the following steps:

1. For each collection c_i in the *NoSQL* database DB_J, a logical source (*xR2RML* semantic view) is extracted using the property ***xrr:logicalSource***.
2. For each document d_k of c_i, a triples map (*tp*) is created. For each *tp*, a subject map is generated, which represents the unique identifier used in all the *RDF* triples generated from it.
3. For each *tp*, a ***predicateObjectmap*** is generated. The predicate is extracted from the input data or the ontology. The object corresponds to the document's value field according to its type:
   – If it is simple, it is mapped to a predicate object and a data property using ***xrr:reference***
   – Else, if it is complex, it is mapped to another triples map and an object property using the ***rr:ParentTriplesMap*** property.

KHG building A *KHG* describes real world entities and their interrelations organized in a hypergraph. It permits the representation of complex structures (classes and their relationships) into a hypernode. Hypernodes are interrelated, using hyperedges. A *KHG* is defined formally in Definition 1.

Definition 1. *KHG: is defined as a tuple* $< N, V, A, S_M, \zeta >$ *with :*

– $N = N_s \cup N_o$ *is the set of the KHG's nodes;* N_s *and* N_o *are the sets respectively of triples' subjects and triples' objects extracted from the set of xR2RML mappings views* (S_M)
– V*: is the set of hyperedges*
– A*: is the set of the directed arcs; an arc is a pair* $< u, v >$ *where* $u, v \in N$
– S_M*: is the set of xR2RML mapping views* (m_i)*, where each* $m_i \in S_M$ *is an hypernode such as:* $m_i = S_n \cup S_a$ *, with* $S_n \subset N$ *and* $S_a \subset A$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

– *ζ is the set of the concepts of the used domain ontology*

The construction of the *KHG* is done via the definition of its:

1. **Entities**: each semantic view is a hypernode (a directed graph which nodes and arcs are respectively the ontology's concepts and properties).
2. **Relations**: relations between the semantic views are the hyperedges of the *KHG*. Two types of hyperedges are constructed:
   – *DBRef* fields from data sources are transformed into hyperedges.
   – Domain ontology's object properties are transformed into hyperedges.

3.2 Reformulation module

The reformulation of a user's query (*U-Q*) into a *MDQ* (Def. 2) is detailed below.

Definition 2. *MDQ is a tuple $Q = (G, S, M, \psi)$, with:*

– *G: is the non-empty set of GROUP BY attributes of the request,*
– *S: is the selection predicate (facultative)*
– *M: is the set of measure attributes which are numeric.*
– *ψ: is the aggregation operator (average, sum, ...).*

Preprocessing Consists of decomposing *U-Q* into words (or set of words) according to their grammatical function in the query phrase using a grammatical resource. A vector of pairs is obtained $\overrightarrow{V} = <p_1, ..., p_n>$, with $p_i = (S_{wi}, f_i)$ and $S_{wi} = \{w_{i1}, ..., w_{ik}\}$ is a set of k words of the decomposed query, where $k >= 1$ and f_i is its grammatical function.

BGPQ schema extraction Is done in two steps:

– **Triples's extraction**: Transforms *U-Q* into a set of triple patterns (S_{tp}). Having $\overrightarrow{V}$ as input, it parses *U-Q* into sub-sentences using the lexical resource. Each sub-sentence turns into a *tp (s, p, o)*, where *s*, *p* and *o* are respectively the subject, verb, and object of the sub-sentence.
– **Triples's aggregation**: Transforms S_{tp} into a *BGPQ* Schema (Algorithm 1). To determine S_{class} (set of classes) and $S_{predicat}$ (set of predicates), for each pair $p_k =< t_i, t_j >$ in S_{tp}, where $k \in [1, |S_{tp}|^2]$, $i \in [1, |S_{tr}| - 1]$ and $j \in [2, |S_{tp}|]$, I_p is the set of common classes between t_i and t_j. If $|I_p| > 0$, non common classes are added to S_{class}, and the respective predicates to $S_{predicate}$. The *BGPQ* Schema is the union of S_{class} and $S_{predicate}$ (line 10).

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

Algorithm 1 Triples Agregation

Input: S_{tp} : Set of triples
Output: $BGPQSchema$
Begin

$S_{class} \leftarrow \{\}$
$S_{predicate} \leftarrow \{\}$
for each $p_k =< t_{pi}, t_{pj} >\in S_{tp}$ **do**
  $I_p \leftarrow \{t_{pi}.subject, t_{pi}.object\} \cap \{t_{pj}.subject, t_{pj}.object\}$
  if $|I_p| > 0$ **then**
    $S_{class} \leftarrow S_{class} \cup \{t_{pi}.subject, t_{pi}.object, t_{pj}.subject, t_{pj}.object\} - I_p$
    $S_{predicate} \leftarrow S_{predicate} \cup \{t_{pi}.predicate, t_{pj}.predicate\}$
  end if
end for
$BGPQSchema \leftarrow S_{class} \cup S_{predicate}$
return $BGPQSchema$

End

Algorithm 2 MDQComponentsExtraction

Input: V: Vector of pairs
$BoolList$: list of boolean operators $(>, <, >=, <=, =, in, \ldots)$
$ListOp$: list of aggregation operators (sum, average, percentage...)
$BGPQSchema$
Output: Ψ, M, S, G
Begin

$max \leftarrow 0$
for each $p_i \in \overrightarrow{V}$ **do**
  for each $w \in p_i.S_{wi}$ **do**
    $m \leftarrow MaxSimAg(w, ListOp, index)$
    if $m > max$ **then**
      $max \leftarrow m$
      $j \leftarrow i$
      $indexMax \leftarrow index$
    end if
  end for
end for
$\Psi \leftarrow ListOp[indexMax]$
$M \leftarrow SearchNumAtt(\overrightarrow{V}[j].S_{wi}, ListNumAtt(BGPQSchema))$
$S \leftarrow SearchPredAtt(\overrightarrow{V}, BoolList, ListAtt(BGPQSchema))$
$G \leftarrow SearchGroupByAtt(BGPQSchema, \overrightarrow{V})$

End

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

BGPQ formulation Algorithm 2 extracts the *MDQ* 's components (Def. 2):

- **Aggregation operator** ψ: for each p_i of $\overrightarrow{V}$, *SimAg()* seeks for the word w of $p_i.S_{wi}$ and the operator of *ListOp*, which are the most similar (Jaccard similarity coefficient). *indexMax* is the index of ψ in *ListOp*.
- **Measure attribute** M: j is the index of S_{wi} containing ψ (line 7). Thus, *SearchNumAtt()* assigns to M the most similar among the numeric attributes of the *BGPQ* schema to the words of S_{wi}.
- **Selection predicate** S **(optional)**: *SearchPredAtt()* searches if a set of words S_w of $\overrightarrow{V}$ contains an operator from *BoolList* and finds the most similar among *ListAtt(BGPQ Schema)*.
- **GROUP BY attributes** G: *SearchGroupByAtt()* returns a set of atomic attributes G, excluding M and S.

3.3 Verification module

In a *MDQ* the measure attributes M and the GROUP BY attributes G must not be on the same dimension hierarchy. The verification is double:

- λ: based on the *KHG*, allows to check the correctness of the request based on the graph of functional dependencies of the grammatical resource (equation 1). The query is correct if $\lambda <> 0$ (a Roll-up if $\lambda > 0$ else a Drill-down).

$$\lambda\left(Att\left(m_i\right), Att\left(g_k\right)\right) = 1 - \frac{\left|Root\left(m_i\right)\right|}{\left|Root\left(g_k\right)\right|} \tag{1}$$

- λ_s: checks the validity of the query against the domain ontology. In equation 2, *Cpt (a)* returns the ontological concept with the attribute a. The query is correct if $\lambda_s <> O$.

$$\lambda_S\left(Cpt\left(m_i\right), Cpt\left(g_k\right)\right) = 1 - \frac{SemanticDepth\left(Cpt\left(m_i\right)\right)}{SemanticDepth\left(Cpt\left(g_k\right)\right)} \tag{2}$$

3.4 Multidimensional SPARQL query treatment module

MD SPARQL query generation is done in three steps:

1. ***SELECT* clause**: followed by the attributes of *BGPQ* excluding G, then ψ followed by M.
2. ***WHERE* clause**: followed by the list of *tp* of the query and if $S <> \emptyset$, the selection predicates are added between parentheses after *FILTER*.
3. ***GROUP BY* clause**: all the attributes of G are added. It should be mentioned that all the attributes of the request are preceded by *'?'*.

Display and storage of results The obtained *MD SPARQL* query is syntactically checked and then executed on the *KHG*. The obtained triples are sent to the display module and the *RDF* store for further use in similar queries.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

4 EVALUATION AND DISCUSSION

To implement the prototype we used: xR2RML[3], Jena API, OWL (Ontology Web Language), OWL API to check syntactically and execute the *MD SPARQL*[4] query, BabelNET API as lexical resource, Stanford dictionary API [5]: as lexical and grammatical resource, and Allegrograph[6]as *RDF* store.

The data collection used is *Patient_survey* (*Data.gov* site[7]) with more than 700,000 records. *Json* Generator tool [8] is used to produce large-scale data with a data schema presented in [8]. These files are loaded in a *MongoDB* database.

4.1 Evaluation of the KHG's completeness

The completeness of information in a *KHG* influences the relevance of data query results. Three completeness metrics are calculated using Sieve[9] and KBQ[10]:

– **Schema Completeness** (***SC***): the rate of ontology's classes and properties in the *KHG*. *SC*=0.97, hence the *KHG* represents a large range of knowledge.
– **Interlinking Completeness** (***IC***): the ratio of interlinked triples. *IC*=0.873, hence the richness of the *KHG*'s properties.
– **Currency Completeness** (***CC***): the ratio of unique triples. *CC*=0.819, so no redundancy.

4.2 Evaluation of the KHG-based MDA

Precision, recall and *F-measure* are used to evaluate the relevance of a set of queries. The average values obtained are respectively 0.82, 0.53 and 0.63. Table 1 reports average *precision* and *recall* for two traditional approaches [9] and ours for which the relevance is improved. In [6], it is reported that after 80% of integrated data sources, these values tend towards 1, when using *KHG*.

Table 1: Comparison of relevance results

Approaches	Average precision	Average recall
Without domain ontology [9]	0.59	0.36
Domain ontology + *NoSQL DB* [9]	0.62	0.52
Our *KHG*-based *MDA* approach	0.82	0.53

[3] https://github.com/frmichel/morph-xr2rml
[4] https://www.w3.org/TR/sparql11-query/
[5] https://nlp.stanford.edu/software/lex-parser.shtml
[6] https://allegrograph.com/
[7] http://healthdata.gov/dataset/patient-survey-hcahps-hospital/
[8] https://json-generator.com/
[9] http://sieve.wbsg.de/
[10] https://github.com/KBQ/KBQ

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19

5 Conclusions and future work

In this paper, a *KHG*-based *MDA* approach is proposed. The idea is to help health experts expressing *MDQ* on multi-source data to improve decision-making. The relevance of the results is improved. In our future work, we intend to study the performance of the approach with real-time treatment and scaling up data size. For the reformulation of queries, deep learning will be used based on our previous works [10].

References

1. Selmi, I., Kabachi, N., Ben Abdallah Ben Lamine, S., Baazaoui Zghal, H.: Adaptive Agent-Based Architecture for Health Data Integration. In: Yangui, S., Bouguettaya, A., Xue, X., Faci, N., Gaaloul, W., Yu, Q., Zhou, Z., Hernandez, N., Yumi Nakagawa, E. (eds.) ICSOC 2019 Workshops, LNCS, vol. 12019, pp. 224–235. Springer, Toulouse, France (2019). https://doi.org/10.1007/978-3-030-45989-5_18
2. Dehdouh, K.: Building OLAP Cubes from Columnar NoSQL Data Warehouses. In: Bellatreche, L., Pastor, O., Manuel, J., Jiménez, A., Aït Ameur, A. (eds.) 6th International Conference, MEDI, LNCS, 2016 vol. 9893, pp. 166–179. Springer, Spain (2016). https://doi.org/10.1007/978-3-319-45547-1_14
3. Chevalier, M., El Malki, M., Kopliku, A., Teste, A., Tournier, R.: Document-oriented Models for Data Warehouses - NoSQL Document-oriented for Data Warehouses. In: Proceedings of the 18th International Conference on Enterprise Information Systems, pp. 142–149. SciTePress, Rome, Italy (2016).
4. Chouder, M. L., Rizzi, S., Chalal, R.: EXODuS: Exploratory OLAP over Document Stores. Inf. Syst. **79**, 44–57 (2019)
5. Gallinucci, E., Golfarelli, M., Rizzi, S.: Approximate OLAP of document-oriented databases: A variety-aware approach. Inf. Syst. **85**, 114–130 (2019)
6. Masmoudi, M., Ben Abdallah Ben Lamine, S., Baazaoui Zghal, H., Archimède; B., Karray, M.: Knowledge hypergraph-based approach for data integration and querying: Application to Earth Observation. Future Gener. Comput. Syst., (2021)
7. Manuél Gómez-Pérez, J., Z. Pan, J., Vetere, G., Wu, H.: Enterprise Knowledge Graph: An Introduction. Exploiting Linked Data and Knowledge Graphs in Large Organisations. Springer (2017)
8. Ait Brahim, A., Tighilt Ferhat,R., Zurfluh, G.: Model Driven Extraction of NoSQL Databases Schema: Case of MongoDB. In: International Conference on Knowledge Discovery and Information Retrieval, pp. 145–154. ScitePress, Vienna, Austria (2019)
9. Radaoui, M., Ben Abdallah Ben Lamine, S., Baazaoui Zghal, H., Ghedira, C., Kabachi, N.: Knowledge Guided Knowledge Guided Integration of Structured and Unstructured Data in Health Decision Process. In: Information Systems Development: Information Systems Beyond 2020, ISD 2019 Proceedings, Association for Information Systems, Toulon, France (2019).
10. Ben Abdallah Ben Lamine, S., Dachraoui, M., Baazaoui-Zghal, H.: Deep Learning-Based Extraction of Concepts: A Comparative Study and Application on Medical Data. Journal of Information & Knowledge Management (2022).
11. G. Schuetz, C., Bozzato, L., Neumayr, B., Schrefl, M., Serafini, L.: Knowledge Graph OLAP. Semantic Web **12**(4), 649–683 (2021)
12. Friedrichs, M.: BioDWH2: an automated graph-based data warehouse and mapping tool. Journal of Integrative Bioinformatics **18**(2), 167–176 (2021)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_19