This is a post-peer-review, pre-copyedit version of an article published in **14th International Conference: Beyond Databases, Architectures and Structures (BDAS 2018)**. The final authenticated version is available online at: https://doi.org/10.1007/978-3-319-99987-6_6

Postprint of: Guminska E., Zawadzka T., EvOLAP Graph – Evolution and OLAP-Aware Graph Data Model. In: Kozielski S., Mrozek D., Kasprowski P., Małysiak-Mrozek B., Kostrzewa D. (eds) Beyond Databases, Architectures and Structures. Facing the Challenges of Data Proliferation and Growing Variety. BDAS 2018. Communications in Computer and Information Science, vol. 928 (2018), pp. 75-89, Springer, https://doi.org/10.1007/978-3-319-99987-6_6

EvOLAP Graph - Evolution and OLAP-aware Graph Data Model

Ewa Guminska and Teresa Zawadzka

Gdansk University of Technology
Faculty of Electronics, Telecommunications and Informatics
Gabriela Narutowicza 11/13, 80-952 Gdansk, Poland
{ewa.guminska, terzawad}@pg.edu.pl

Abstract. The objective of this paper is to propose a graph model that would be suitable for providing OLAP features on graph databases. The included features allow for a multidimensional and multilevel view on data and support analytical queries on operational and historical graph data.
In contrast to many existing approaches tailored for static graphs, the paper adresses the issue for the changing graph schema.
The model, named Evolution and OLAP-aware Graph (EvOLAP Graph), has been implemented on a time-based, versioned property graph model implemented in Neo4j graph database.

Keywords: Graph Database, Multidimensional Data Model, OLAP, Neo4j, Property Graph, EvOLAP Graph

1 Introduction

The changing nature of data into high-volume, high-variety and high-velocity information assets has given a rise to the development of new data technologies used in up-to-date information systems. Among all, a graph data model and graph databases have gotten much of interest and popularity.

The graph data model is designed to handle highly interconnected data and to investigate connections between them. Therefore, it is widely used in systems which explore the structural information coming from the relationships between entities. Consequently, graph databases are sometimes becoming an alternative to relational databases which, in many cases, are insufficient and limited.

Many businesses and entrepreneurs are interested in expanding analysis based on new data types, in particular on graph data. Moreover, as On-Line Analytical Processing (OLAP) technology is already widespread, enterprises are interested in expanding their OLAP analysis on the new types of data.

Combining power of OLAP and graph database technologies can result in better analyses which, in consequence, can lead to more accurate decisions. However, in traditional data warehouses, the topological information of relationships between the data is defined in the phase of the dimensional model design, and therefore, cannot be simply adapted to a flexible nature of graphs. Lack of this

flexibility can lead to the limits of potential insight. This is one of the important reasons, why we can observe the growing interest in executing OLAP analyses directly on graph data, without the need for building traditional data warehouse infrastructure and using of complex mapping processes.

Therefore, the studies described in this paper address the need of enhancing graph databases with OLAP features - ability to issue analytical queries directly on the graph database without the need of building dedicated traditional data warehouses.

The objective of this paper is to propose a graph model, based on a property graph model, that contains more than one type of the entities or relationships (heterogeneous graph). Such model is suitable for providing OLAP features on graph databases (OLAP-awareness). The features support analytical queries on operational and historical data and address changes in the graph scheme (evolution-awareness).

The rest of the paper is organised as follows: Chapter 2 analyses related works and existing approaches that enrich graph databases with OLAP features, while Chapter 3 introduces formal EvOLAP Graph concept, discusses the architecture, evolution-aware and OLAP-aware functionalities. Next, Chapter 4 describes the implementation of the EvOLAP Graph and evaluation of the work. Finally, Chapter 5 comments on the study results and draws conclusions.

2 Existing Approaches

There has been growing interest in combining graph databases with Business Intelligence technologies in recent years. Many studies have been undertaken aiming at inclusion in data warehouses unstructured data types such as document-oriented databases [1], text [2, 3], XML [4] or column-oriented databases [5]. Some approaches have also considered graph databases [6] and graph data representation in Semantic Web data [7, 8]. Currently many researches focus on new methods of carrying out OLAP analysis directly on graph databases and in the following sections we focus only on such approaches. Morover, from the whole spectrum of existing approaches we briefly introduce the ones that accomplish at least 1 of the 3 main assumptions of EvOLAP Graph named in Introduction.

2.1 Graph OLAP

The first concept called Graph OLAP [9] focuses on OLAP inside singular, homogeneous and static graphs. It presentes OLAP analysis on graphs by introducing the concept of *Informational OLAP* and *Topological OLAP*.

In *Informational OLAP* all queries are executed against the original graph. The main aim is to deliver snapshots that focuse on the non-topological information contained in nodes and edges of the graph.

By contrast, in *Topological OLAP* the original graphs topological structure might be altered. The main aim is to deliver the topological information. Thus,

the aggregation functions are used to create new graph structure which delivers a summarised node of multiple source nodes.

Moreover, the idea which assumes that an aggregated graph might be thought of as a measure was presented. Depending on the type of dimension an aggregation is created from - informational or topological - the *I-Aggregated Graph* or *T-Aggregated Graph* is computed.

Graph OLAP is one of the first concepts which face the problem of developing OLAP on graph databases. However, the main weakness in this approach is that authors assumed static graphs, thus they did not examine problems coming from the nature of graph data as dynamic structure of the model and volatile data in data sources.

2.2 Graph Cube

In the next concept, to facilitate OLAP analysis on graphs, the concept of Graph Cube [10] has been described. The model focuses on OLAP inside a singular large multidimensional network. Building Graph Cube model requires forming all possible aggregations of the network.

Moreover, a new OLAP query model has been suggested and two types of applicable queries have been distinguished: *cuboid* queries, corresponding to traditional OLAP queries, where an output is a simple aggregation of the underlying graph and a new kind of queries, *crossboid*, where the query crosses several spaces of the network, thus calculation of the result uses more than one cuboid.

However, the main noticed limitation of Graph Cube is the usage of graph model that does not have either properties or labels, which considerably restrict application of this approach. Again, the problem of graph's changes in time was not considered.

2.3 HMGraph OLAP

The first approach that addresses the problem of heterogeneous data networks is HMGraph OLAP [11]. It has been noticed that informational and topological aspects may not be fully suitable to handle heterogeneous graph analysis and many graphs are not appropriate for drill-down and roll-up operations.

Therefore, to overcome these shortcomings HMGraph OLAP framework enriches graphs by more aspects and new types of operations. A concept of *Entity Dimension* has been defined as the set of entity attributes. Additionally, new operations - *Rotate* and *Stretch* - has been introduced to enable mining implicit knowledge within the graph. *Rotate* operation changes the node with more than two edges into new edges between new nodes generated from the original edges. Following, *Stretch* operation changes an edge into a node and adds edges between the new nodes and their initial endpoints.

Morover, the concept of *Relation Meta Path* has been introduced. A meta relation path is a composite relationship between 2 nodes consisting of a sequence of relations between them.

This approach is one of the first that enables multidimensional and multilevel analysis. However, the main shortcoming of this framework is that it has been developed with immutable simple graph models and only a star schema for a data warehouse has been considered. The problem of graphs changes hasnt been studied.

2.4 Analytics-Oriented Evolving Graph

Another interesting work [12] focuses on a graph metamodel and proposes the graph model which is analytics-aware and deals with versions of the data e.g. different language versions of an entity.

The model contains 3 different types of nodes and 4 types of edges. First, *entity nodes* representing graph entities followed by *evolving nodes* that describe attributes of *entity nodes* which changes must be tracked. The *value nodes* hold the value of related evolving nodes. The associations between *entity nodes* are described by *entity edges*. *Hierarchical edges* depict an aggregation relationship. *Evolving edges* denote a composition relationship and the last ones versioning edges represent an association between *value nodes* and *evolving nodes*.

The model supports analysis thanks to the two new data structures - *analytic hypernodes* and *classes*. *Analytic hypernode* is a subgraph made from an *entity node* and all related *evolving* and *value nodes* with all edges between them. A *class* groups *analytic hypernodes* whose *entity nodes* belong to the same label.

Indisputable strength of this model is node versioning. However, it does not address other evolutions on graphs, like evolutions that change graphs over time.

The Table 1 compares the presented solutions with Evolution and OLAP-aware Graph (EvOLAP Graph) which is the one of the first solutions that takes into consideration the dynamic nature of graph data and focuses on evolving, volatile and heterogeneous graphs.

Table 1. Comparison of existing approaches with EvOLAP Graph main features

Approach	OLAP -awareness	Evolution -awareness	Heterogeneous graphs
Graph OLAP	+	-	-
Graph Cube	+	-	-
HMGraph OLAP	+	-	+
Evolving Graph	+	+/-	+
EvOLAP Graph	+	+	+

3 EvOLAP Graph

The key assumption of EvOLAP Graph concept is the combination of the two qualities - *evolution-awareness* to handle the flexible nature of the graph data and *OLAP-awareness* to provide core OLAP features in graph databases.

Volatile data and a flexible schema-less dynamic structure of graphs make the analysis of the graphs' data and structure over time especially difficult. Yet, such analysis is characteristic for data warehouses, where historical data are processed. To enable similar analysis on graph databases, it is necessary to enhance the graph data model by evolution-awareness (Fig. 1). The crucial issue which must be addressed is data changes along with structure changes. As the structure information held in a graph model is the model's principal advantage over other data models, tracking this topological information changes in time can potentially give an additional insight that is not easily available in standard data warehouse solutions. Therefore, EvOLAP Graph makes use of a *time-based versioned graph* [13] - the one capable of handling time-based versioning of both: the data and the graph structure.

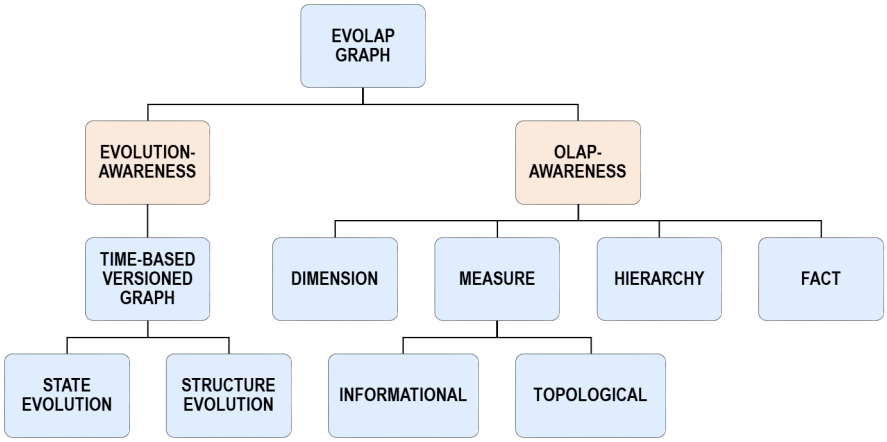

Fig. 1. Architecture overview of EvOLAP Graph concept

The second feature - OLAP-awareness - gives a special analytical view on a graph model ensuring that OLAP queries might be answered successfully. It enriches the graph model with such features as dimensions, measures, hierarchies and facts the notions well-known from data warehouses (Fig. 1). Such view of the graph model rationally justifies conducting OLAP analysis directly on the graph data model.

3.1 Evolution and OLAP-aware Graph Metamodel

EvOLAP Graph, introduced in this paper, is defined as a single graph $G = (V, E, L)$, where $V = \{V_s, V_i\}$ is the set of nodes, $E = \{E_l, E_{str}, E_s, E_h\}$ is the set of edges and L is the set of labels of all nodes and edges. The metamodel of EvOLAP Graph is shown in Fig. 2.

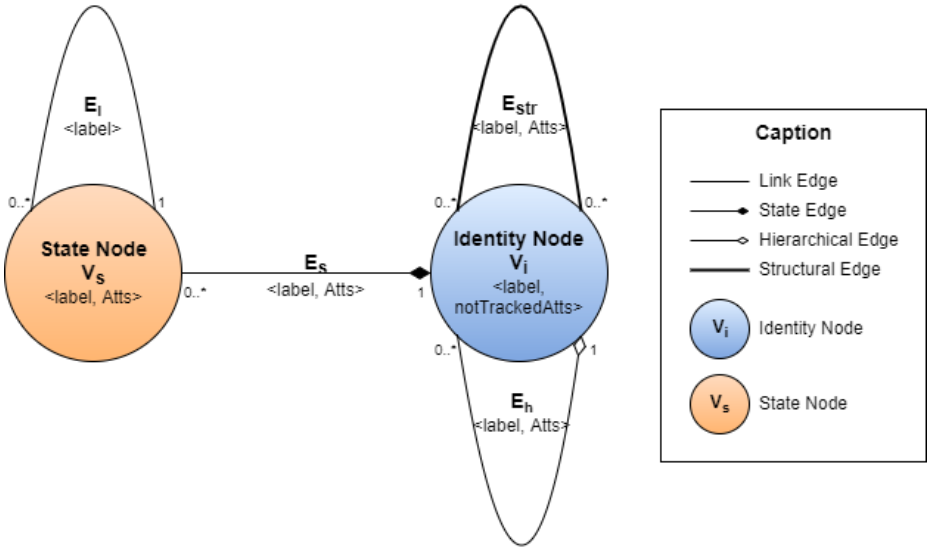

Fig. 2. EvOLAP Graph metamodel

In order to provide time-based tracking on graph databases for evolution-awareness, the approach *separating structure from the state* of Ian Robinson [13] has been implemented. The idea is to allow for independent versioning of graph topology and state. Therefore, a distinction between structure and state parts of the graph is made. For that reason, Robionson in his work proposed a terminology, that EvOLAP Graph adopted.

The structural parts are *identity nodes* - V_i and *structural edges* - E_{str}, whereas state parts are called *state nodes* and *state edges* - V_s, E_s respectively. An *identity node* is a node that identifies an entity and locates it in the graph topology. It does not contain any attributes that would represent its state. For this purpose, *state nodes* are used. They are linked to an identity node by *state edges*, which have two additional attributes *FROM* and *TO*, indicating the time period when this snapshot of an entity's state is valid. *Structural edges* that associate identity nodes accommodate these two properties as well. Optional *link edges* - E_l - link successive *state nodes* associated to the same *identity node*.

Additionally, EvOLAP Graph introduces *hierarchical edges* and *hypernodes*. A *hypernode* is a metanode that consist of *idetntity node* with all connected *state*

nodes. *Hierarchical edges* E_h associate levels of a hierarchy which is abstracted as *identity nodes*.

The mechanics of the model allows for addition and modification of graph strcture - never deletion. If an entity is added to the graph, the representing *value node* and *structural edges* are created. Further modifications result in addition of new *state nodes* representing changes of the entity over time. Deletion is treated in the same way as modification - a new state of the entity is stored in the graph.

3.2 Evolution-awareness

State and structure evolution Evolution-awareness of EvOLAP Graph allows to analyse changes of the graph state and structure over time.

The case of modification of the state is shown in Fig. 3. Here, on 01/01/2018 the price of the book changed, thus the old state relationship's property *TO* was set, the *Bookstate* state node was added and associated with the identity node. The fact of archiving the state relationship is illustrated by the dashed line.

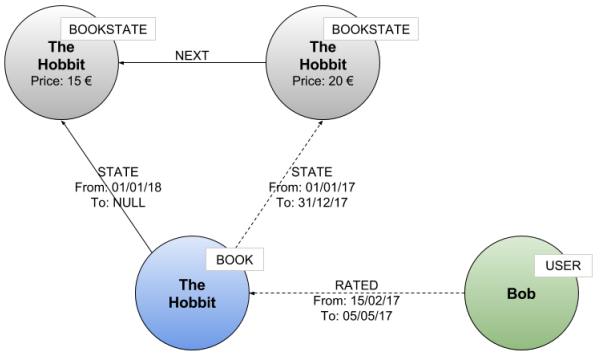

Fig. 3. Time-based versioned graph

An example of modification of the graph's structure is shown when Bob cancels his "The Hobbit" book rating. In the version-free graph, the edge between "The Hobbit" and Bob would be deleted, however in EvOLAP Graph that is not an option. The relationship *RATED* is archived by changing the value of *TO* attribute.

Likewise, deleting of a node can be done by archiving its current state and all relationships that an identity node is a part of. The case of adding a node or relationship is trivial one.

In that way, by adopting *separating structure from the state* approach, EvOLAP Graph handles some analytical queries exemplified beneath - that in the data warehouses would typically take dimension *Time* into account:

– How has the price of the product P changed in region R1 last year in comparison to region R2?

– What is the evolution of the shortest path between the user U and the product P over time?

It is worth to notice, that the solution presented here is an answer for *Slowly Changing Dimension* (SCD) issue in traditional data warehouses. SCD is a situation where the historical data stored in a data warehouse are subject for the change [14]. The method proposed in EvOLAP Graph is an adapted SCD type 2 where the old data are preserved and the new ones are added. Both versions of data are updated with the information on validity periods.

3.3 OLAP-Awareness

In this chapter the core data warehouse terms - dimensions, measures, facts and hierarchies - are presented for EvOLAP Graph. However, the proposed term mapping is not unambiguous in terms of a graph model. The terms become apparent in the moment of formulating a specific query. Therefore, in EvOLAP Graph, elements of a graph model as nodes, relationships and attributes can be defined as mapped dimensions, measures or facts only during the query lifetime. For example, in context of the query "What are the average ratings for all Tolkien's books?" nodes Books and Authors function as dimensions, the attribute "Stars" of the relationship RATED functions as a measure and relationship RATED serves as a fact.

Dimension A distinction of dimensions in the data warehouse enables a multidimensional point of view on measures for analysis. In graph databases, a label of a node can be thought of as a dimension and one point of view on data. In EvOLAP Graph where the structure and the state of the entity are separated, both labels: of an identity node and of its state nodes are the parts of one dimension. Fig. 4 shows a portion of a graph which contains nodes of four labels: *Book*, *Author*, *User* and *Bookstate*. Thus, three dimensions can be distinguished: *Authors*, *Books* and *Users*. Hence a book entity is versioned and its state and identity are separated, both labels *Book* and *Bookstate* belong to the *Books* dimension.

Dimension's attributes can be only the attributes of the node associated with them. For example a dimension *Book* could include attribute *literary period*, dimension *Author* - an attribute *nationality*.

As in the data warehouses as well in EvOLAP Graph, a dimension has many dimension elements. Those of them that belong to the dimension include all hypernodes of the same type in the graph, where a hypernode is defined as an identity node with all its state nodes. The hypernode *Hobbit* signalized by the orange rectangle (Fig. 4) is an identity node with all its state nodes. Elements of the dimension *Books* consists of all hypernodes of the type *Book* which are indicated by the blue rectangle.

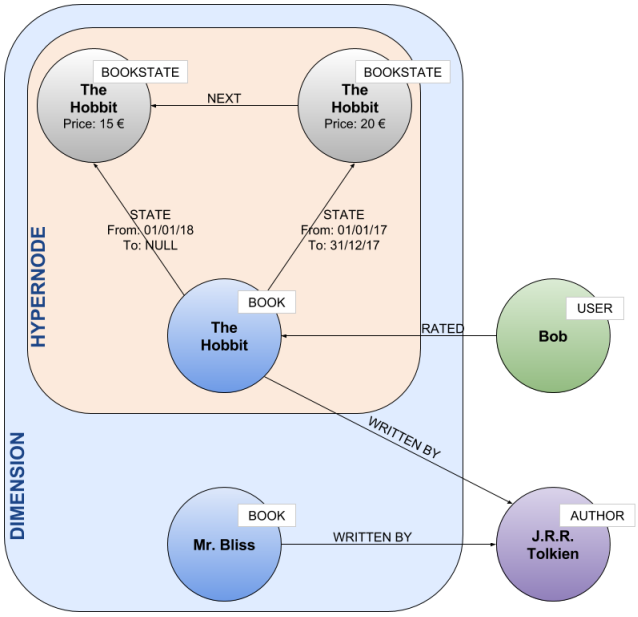

Fig. 4. Dimension definition in EvOLAP Graph

Measure A measure is a metric that is used to analyse a subject. In EvOLAP Graph two types of measures are conceived.

Informational measure The first one called informational measure is the one where a numeric metric is contained in the attribute of the node or in the relationship of the graph. For example, the value of the attribute "Price" in the node "Book" describing a price of a book, or a value of the attribute "Stars" in RATED relationship describing a rating given to a book by the user, are informational metrics.

Topological measure The second metric, a topological measure is one which comes from the graph theory or the graph's structure analysis as a network density, a degree of a node, the shortest path between nodes, and the like. For example, the shortest path between user Ron and book "The Hobbit" or the number of relationships RATED coming from a node "The Hobbit" are topological measures.

It follows then that informational measures are comparable to traditional ones used in data warehouses. However, an overwhelming advantage of EvOLAP Graph is information contained in the structure of the graph. To get access to it, the topological measures must be used.

Fact A fact in EvOLAP Graph cannot be explicitly observed and defined in the data model structure. In data warehouses, a purpose of a fact table is to

be a repository for the measures [14]. As the measures in EvOLAP Graph can be distributed over the whole graph, the fact can be concealed in the entire graph. On the logical level, the fact describes an event in the physical world i.e. a purchase. Likewise, in EvOLAP Graph an event as a fact can be present. However, it can be modelled either as a node or a relationship, i.e. as a node describing a purchase or as a relationship PURCHASED.

Hierarchy In data warehouses creating hierarchies in dimension and fact tables enables multilevel point of view. It is done by defining drill-down and roll-up paths which focus on more detailed or summarized views. These views are created by aggregation of measures on hierarchy levels [15].

The very helpful example is for instance a dimension *Books* with a hierarchy defined as (all) → (category) → (genre) → (book). This hierarchy might be used for example to check the popularity of all books of the same genre. The measure popularity will be aggregated accordingly.

To create the hierarchy in EvOLAP Graph a special type of a relationship - hierarchical edge - must be added. This relationship indicates a start of a hierarchy path and a direction of increasing detail level. (Fig. 5).

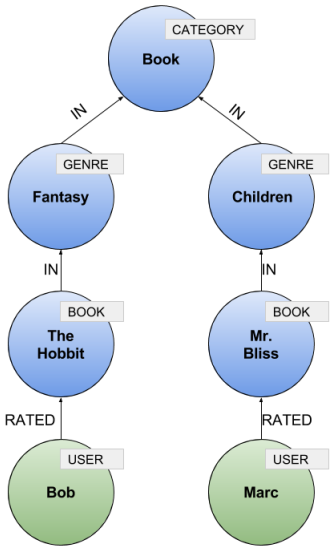

Fig. 5. Example of a hierarchy in EvOLAP Graph

4 Experiments

The EvOLAP Graph model has been evaluated in terms of: functionality by checking analytical capabilities, and performance by presenting execution time of queries for increasing graph size.

All experiments were conducted on laptop computer with 8 GB RAM, Intel Core i5-3210M, running Windows 10. The Neo4j graph database (version 3.2.3) has been configured to not to cache query execution plans by setting *dbms.query_cache_size* property to 0 in the configuration file.

4.1 Functional Evaluation

To check EvOLAP Graph analytical capability, the list of exemplar analytical queries has been run. The scope of these queries includes informational and topological operations, cascaded and not cascaded access patterns, analysis of the graphs state and its structure over time. All predefined analytical queries were implemented [16] and executed.

Informational queries

1. Who is the author of The Hobbit?
2. What are the average ratings for all Tolkien's books?
3. What is the difference between the average ratings of standard users and premium users (users who pay for an annual subscription) of Fantasy and Horror books?
4. What is the trend for the number of premium users between the last year and 2 years ago? Is BookLovers (a website for readers and book recommendations) attracting more or fewer premium users over time?
5. What has been the average rating history of "The Hobbit" year by year?
6. What books can be recommended to Bob? What books have been rated by other users who have rated the same books as Bob has giving those good ratings?
7. Which genres has Bob been interested in?

Topological queries:

8. How many out of all the users, have not rated any book yet?
9. How many friends every user has on average?
10. What is the shortest link between Eve and "The Hobbit" book?
11. How many ratings of Fantasy books did premium and standard users give last year on average?
12. What is the trend of the number of new ratings given for each genre in Book category over last two years?
13. What is the evolution of the shortest path between Eve and Ron over time?
14. What is the evolution of the number of Ron's friends over time?

The basic graph data set that was used contained 50 nodes (30 identity nodes and 20 state nodes), 169 properties and 66 relationships. The maximum degree of the graph was 8 and the minimum degree was 1. It was built in compliance with the EvOLAP Graph model - it separated the structure and the state of the graph data, it comprised the time-based versioning model elements for each

evolving relationships and nodes. Additionally, it applied the EvOLAP Graph hierarchy concept by abstracting hierarchy nodes from attributes and indicating a hierarchy direction by the hierarchical relationship.

The results showed that EvOLAP Graph can answer questions that are common in use in data warehouses, and can answer topological questions that are much easier and more natural for graph data models. Queries: 4, 5, 11 - 14 would not be achievable in traditional graph databases without time-based versioning solutions. Additionally, correct results were obtained from queries that would be particularly challenging to implement in data warehouses or relational databases (queries 6, 7, 8 - 14).

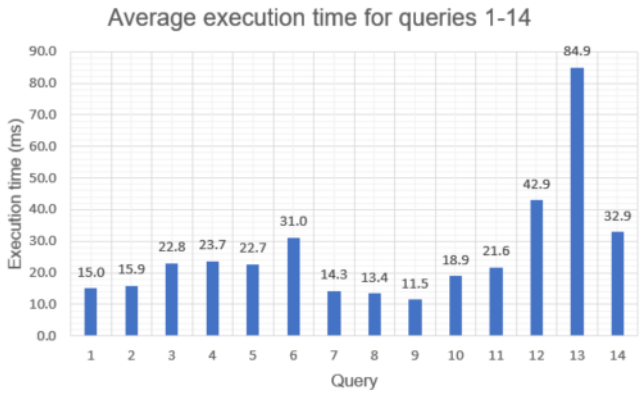

Fig. 6. Average execution time for queries 1-14

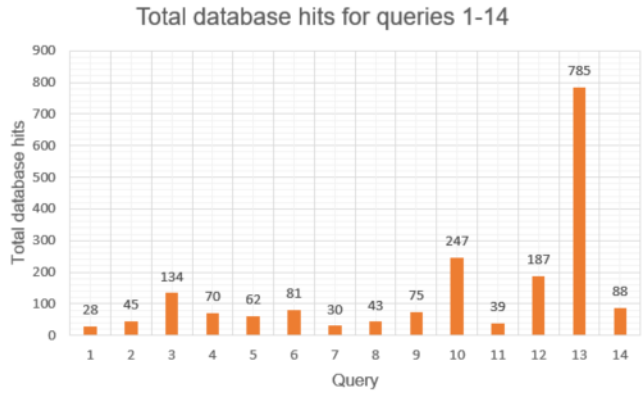

Fig. 7. Total database hits for queries 1-14

All executed times as well as total database hits are shown in diagrams below (Fig. 6, Fig. 7). It is easy to notice, that there is no major difference between informational (1-7) and topological (8-14) queries in terms of execution time and database hits. However, query 13 which explores the evolution of the shortest path over time has considerable higher execution time.

4.2 Performance Evaluation

In the second step, the performance of the model has been evaluated in terms of increasing graph size. The evaluation of the scalability of EvOLAP Graph can reveal what is the expected performance of the proposed solution on bigger data sets or on highly volatile data sets.

The experiment checked an average query execution time for various datasets to test performance trend according to the graph's size. Generated datasets differed by their size from 50 to 1 000 000 nodes in each graph. A method used focused on ensuring comparable graph density between all datasets.

All fourteen queries have been executed on all datasets. The results are shown in diagram (Fig. 8).

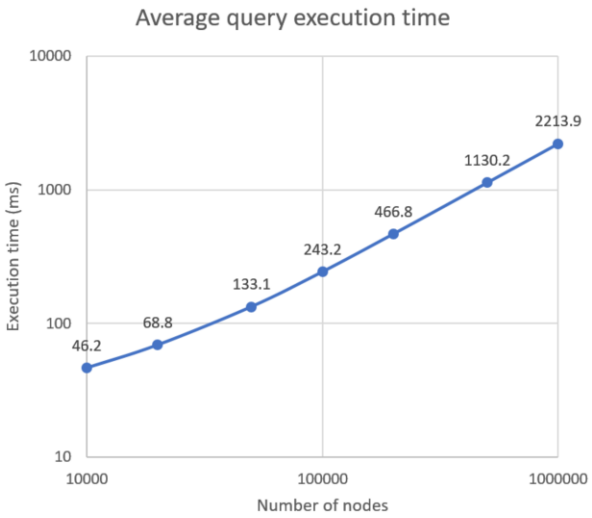

Fig. 8. Average query execution time for increasing EvOLAP Graph's size

It is easy to notice that the execution time increases nearly linearly with the size of the graph and this observation holds true for informational and topological queries. This conclusion means that for graphs of a bigger size the performance of a database will also decrease linearly.

5 Conclusions

The presented model named Evolution and OLAP-aware Graph (EvOLAP Graph) is one of the first approaches that propose the method of enabling OLAP analysis on evolutionary, volatile, dynamic graphs. In contrast to many of the existing approaches, it does not use static or homogeneous graphs only. It facilitates a multidimensional and multilevel view of data by defining star schema notions such as dimensions, measures, facts and hierarchies. The developed model is built on the time-based versioned graph that is built upon the property graph model.

The results of the experiments proved that EvOLAP Graph enables OLAP features on heterogeneous and evolving graphs. It is able to analyse over time changes in the graph's state and structure. Therefore, the analyses are enriched with the topological information which would be unachievable in traditional data warehouses. The developed model enables to get answers from a variety of analytical queries, including questions that would be particularly difficult to handle in traditional graph databases or data warehouses. The experiment regarding the performance showed that there is a linear increase of average query execution time with increasing dataset size measured in the number of nodes.

One source of weakness in this study is the method used for the time-based versioning of the graph. The method applies to the graph *separating structure from the state* approach which leads to the expansion of EvOLAP Graph. Additionally, EvOLAP Graph used as a type of a data warehouse handles historical data, which naturally leads to the growth of the stored data over time. Thus, the consequences such as a lowered performance of reads, a higher cost of writes and additional disk storage requirements are the major limitations of EvOLAP Graph.

For that reason, further research works focus on extending EvOLAP Graph by an additional innovative data structure that is designed to improve analysis performance and lower the storage requirements - EvOLAP Cube. At the same time, EvOLAP Graph should be tested in a production environment.[**?**]

References

1. Chavalier, M., Malki, M.E., Kopliku, A., Teste, O., Tournier, R.: Document-oriented data warehouses: Models and extended cuboids, extended cuboids in oriented document. In: 2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS). (6 2016) 1–11
2. Park, B.K., Song, I.Y.: Toward Total Business Intelligence Incorporating Structured and Unstructured Data. In: Proceedings of the 2Nd International Workshop on Business intelligencE and the WEB. BEWEB '11, New York, NY, USA, ACM (2011) 12–19
3. Liu, X., Tang, K., Hancock, J., Han, J., Song, M., Xu, R., Manikonda, V., Pokorny, B.: SocialCube: A Text Cube Framework for Analyzing Social Media Data. In: 2012 International Conference on Social Informatics. (12 2012) 252–259

4. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: A multi-dimensional framework for graph data analysis. Knowledge and Information Systems **21**(1) (2009) 41–63
5. Dehdouh, K., Bentayeb, F., Boussaid, O., Kabachi, N.: Using the column oriented NoSQL model for implementing big data warehouses. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'15) (2015) 469–475
6. Liu, Y., Vitolo, T.M.: Graph Data Warehouse: Steps to Integrating Graph Databases Into the Traditional Conceptual Structure of a Data Warehouse. In: 2013 IEEE International Congress on Big Data. (6 2013) 433–434
7. Nebot, V., Berlanga, R.: Building Data Warehouses with Semantic Web Data. Decis. Support Syst. **52**(4) (3 2012) 853–868
8. Laborie, S., Ravat, F., Song, J., Teste, O.: Combining Business Intelligence with Semantic Web: Overview and Challenges. In: INFORSID. (2015)
9. Chen, C., Yan, X., Zhu, F., Han, J., Yu, P.S.: Graph OLAP: Towards online analytical processing on graphs. In: Proceedings - IEEE International Conference on Data Mining, ICDM. (2008) 103–112
10. Zhao, P., Li, X., Xin, D., Han, J.: Graph cube: on warehousing and OLAP multi-dimensional networks. In: SIGMOD - Proceedings of the 2011 International Conference on Management of Data, New York, NY (2011)
11. Yin, M., Wu, B., Zeng, Z.: HMGraph OLAP: A Novel Framework for Multi-dimensional Heterogeneous Network Analysis. In: Proceedings of the Fifteenth International Workshop on Data Warehousing and OLAP. DOLAP '12, New York, NY, USA, ACM (2012) 137–144
12. Ghrab, A., Skhiri, S., Jouili, S., Zimányi, E.: An Analytics-Aware Conceptual Model for Evolving Graphs. In: Proceedings of the 15th International Conference on Data Warehousing and Knowledge Discovery - Volume 8057. DaWaK 2013, New York, NY, USA, Springer-Verlag New York, Inc. (2013) 1–12
13. Robinson, I.: Time-Based Versioned Graphs. http://iansrobinson.com/2014/05/13/ time-based-versioned-graphs/ Accessed: 2017-09-07.
14. Kimball, R., Ross, M.: The Data Warehouse Toolkit, The Definitive Guide to Dimensional Modeling. (2013)
15. Malinowski, E.: Advanced Data Warehouse Design: From Conventional to Spatial and Temporal Applications. Springer-Verlag, Berlin, Heidelberg (2009)
16. Guminska, E.: Analytical Dimensional Model in Graph Databases. Master's thesis, Gdansk University of Technology (2017)