Effective Visualization of Data Structures in Graph Databases

Adam Dudáš * and Adam Kleinedler

Department of Computer Science, Faculty of Natural Sciences, Matej Bel University, Banská Bystrica, Slovakia
Email: adam.dudas@umb.sk (A.D.); adam.kleinedler@student.umb.sk (A.K.)
*Corresponding author

***Abstract*—To efficiently store data where the relationships between individual objects are essential, the use of a graph database model is recommended. After storing the data, it is necessary to further analyze it using statistical methods or visualize it within the context of exploratory data analysis. Such visualization is crucial for understanding the structure and content of the database. However, commonly used visualization tools often fall short in terms of interactivity and effectiveness. The main objective of the presented work is the design and implementation of a novel model for the visualization of data structures stored in graph databases with the use of two natural graphical models—the standard topological layout of the database and the so-called clustered layout of a graph database. The presented graphical models are focused on interactive visualization, mainly scaling of visualization and development of database objects, and principles of effective visualization. Implementation of the proposed approach was evaluated via case studies on three model graph databases of various sizes—Messaging database (16 objects, 16 relationships), Library database (16 objects, 32 relationships) and Movie database (171 objects, 253 relationships). Compared to the standard Neo4j tool for the visual representation of property graphs in graph databases, the proposed model presents improvement in terms of the number of visualization models, effectivity of the visualization, and development of objects in the visualized database.**

***Keywords*—visualization, graph databases, clustered visualization layout, data representation**

I. Introduction

Graph databases are complex, non-relational databases focused on the effective representation of relationships between entities stored in the database. This leads to their frequent use in the sets of data which are interconnected with relationships of importance [1]. Similar to other data models, the graph database model is based on three fundamental properties [2] - the structure in which the data is stored, the language used to manipulate the data, and a set of rules that ensure data integrity and correctness. In the graph database model, the structure of the data consists of a property graph, which can be described as previous study [3]:

$$G = (V, E), E \subset V^2 \quad (1)$$

where V is a set of nodes, while each node represents a database entity with attributes and their respective values stored in the form of *key:value* pairs similar to other non-relational databases. Each node has its type which labels.

The significance of the object and prompts what attributes will be recorded for the object. E denotes a set of edges that can be oriented and labeled. These edges in the property graph illustrate relationships between individual entities in the database (nodes). An example of a property graph is presented in Fig. 1.

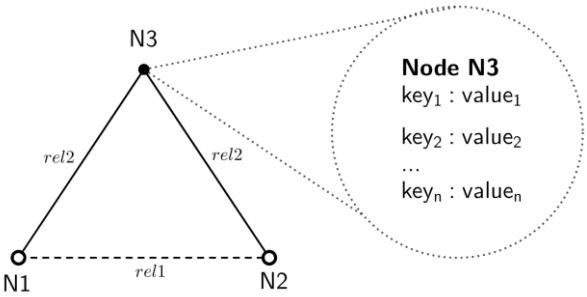

Fig. 1. Example of the structure of data in graph databases—there are two types of objects (nodes) in the database and two types of relationships (edges) between these objects.

Following storage of data in a graph database, subsequent work with the data and their analysis is natural. The complex data structures present in a database need to be examined and visualized for experts to discover trends, relationships and dependencies in the data model. Visualization of data structures plays a pivotal role in comprehending the structural aspects and content encapsulated within a graph database. Nevertheless, conventional visualization tools are often insufficient in terms of interactivity and effectivity, which necessitates advancements in visualization techniques.

The main objective of the presented work is the design and implementation of a model for the effective visualization of data structures in graph databases. The proposed model uses two different types of graph visualization layouts:

Manuscript received February 15, 2024; revised April 7, 2024; accepted April 23, 2024; published August 20, 2024.

- Standard, topological layout, where the data is organized in the form of a property graph as described in Fig. 1. Since this graphical model visualizes all the objects of the graph database, it is most commonly used in graph database visualization problems. Yet, when working with a large graph database containing hundreds of objects interconnected by hundreds or thousands of relationships, the readability of such visualization is significantly lowered.
- Clustered layout, in which identical types of data objects and relationships between objects are clustered. This visualization model can be used in descriptive and exploratory data analysis as a form of familiarization with databases, which is in many cases a key step of data analysis.

Both of these layout types are focused on interactive functionalities of visualization models—mainly scaling of the view of the property graph of the database and development of data objects—while adhering to principles of effective visualization, such as elimination of distortions, proper scaling and labeling and so on.

The main contribution of the article can be summarised as the proposal of the novel-cluster-model for graph database visualization, implementation of this model, and experimental evaluation of the proposed model on three graph databases with comparisons to the standard Neo4j tool for visual representation of property graphs in graph databases.

The body of the presented work is structured into four main sections. In Section II, we analyze works related to the research presented in this article, while focusing on graph databases, their use in data analysis, and the visualization of this type of database. Section III contains design and implementation notes for the proposed models of visualization of data structures in graph databases, which are, then, experimentally evaluated in the form of case studies presented in Section IV. Section V presents the conclusion of the work and proposes areas for future work in the graph dataset visualization.

II. Related Works

Even though relational databases provide high-quality support for structural data management, modern systems need to be able to work with large volumes of data characterized by their variable structure [4, 5]. When storing datasets consisting of heterogeneously structured subsets, the use of non-relational databases in the form of document databases, *key:value* databases, or graph databases is appropriate [6, 7]. Since the presented work focuses on the visualization of graph database structures and substructures, this section comprises works related to graph database visualization and data analysis within the context of graph databases.

Ferilli *et al.* [8] describe a framework and online platform for the internet-based knowledge graph definition, population, and exploitation based on the labeled property graph model. Among other topics, the study identifies three types of interesting graph database visualizations—data visualization, model visualization, and data-to-model visualization. In terms of the work, data visualization refers to the use of graphical representations to enhance data comprehension. Model visualization specializes in illustrating data models, including schemas and ontologies, providing insights into underlying data structures. And, data-to-model visualization refers to schema extraction over resource description framework.

One of the most interesting problem-specific applications of graph database stores is presented in the work [9]. The authors propose a design and implementation of a platform for a universal biomechanical application aimed at handling the data regardless of its type or metadata. The primary focus of the study is to implement a system with integrated, standardized visualization elements that can be used in any sensor-based biomechanical application. Interestingly, the authors point out the high possibility of oversaturation of graph database visualization for even medium-sized graph databases and define the need for the development of new graph database visualization layouts that solve this issue.

On the other hand, Orlando *et al.* [10] elaborate on the topic of visualization in the context of temporal property graph databases. To use the full potential of such databases, this model requires visualization tools that allow navigating graph data across time. The authors of the study propose a framework for temporal property graph visualization based on a data model and query language for temporal graphs implemented over Neo4j. The proposed model allows editing and running of queries, presentation of results, and navigating such results across time.

The visualization of graphs or networks is also the main objective of the work presented in [11]. The authors of the work focus on network security visualization and propose a framework with the main functions of its constituent elements for this purpose. Based on the analysis of network security data structure, the authors design and implement a network security data organization method based on a graph database.

Finally, Mueller *et al.* [12] focus on mapping mechanisms for the translation of relational structures to the form of graph databases. The main objective of the work is to point out the challenges of such mapping and the commonly overlooked problems presented in this task. While trying to resolve these issues, the authors present an improved user interface for working with graph databases, which brings certain no-code practices to graph databases and ensures the visualization of graph databases based on the Neo4j engine.

III. Visualization of Data Structures in Graph Databases

While visualizing the data structures present in a graph database, one needs to consider several criteria for visualization models. For the purposes of the proposed model, we focus on potential descriptive and exploratory analysis of the data stored in the database and, therefore, the criteria of visualization effectiveness and interactivity are crucial.

Interactivity in visualization models refers to the functionality that allows users to obtain more information

from a visual display than by merely viewing it. With the use of interactivity in visualization, the user of the model gains access to interactively request more contextual data, select parts of the data for further data analysis, investigate different ways of configuring computational methods and alternative models for more informed analysis, and so on [13].

The most basic of considered interactive functionalities for the proposed model are:

- Scaling of the visualization. Since the model visualizes a property graph from the database, which may contain hundreds or thousands of nodes and edges, it is essential to scale the view of this graph. Basic scaling by zooming in and out of the visual representation of the database is essential for the presentation of the structures of the database.
- Development of data objects. Graph databases consist of structures within structures. Specifically, the whole database is composed of nodes and edges while each node consists of several *key:value* pairs. It is natural that when visualizing such a database, the model cannot effectively present both of these structures. Therefore, the need for the development of node (visualization of node structures and values after their selection) in the property graph of the database is needed.

In information and data visualization the effectivity of the method itself is a property that can be described as a sum of the following criteria [14–16]:

- Maximization of the ratio between used color and data-since the objective of the visualization model itself is to visualize the data structures in a graph database, in an ideal scenario, the visual model contains a minimum of other graphic elements (a background color, a distinctive grid, and so on). Maximizing the ratio of color and data is essential, especially when visualizing large property graphs that can potentially merge or be rendered by very small nodes and edges.
- Elimination of distortions. When visualizing data structures, it's important to address two types of distortion—data distortions and image distortions. Since the proposed model follows the interactivity of the visualization in the form of zooming options, the distortions and misleading factors should be minimized or removed altogether.
- Use of proper scales and labels. For the purposes of visualization of graph database structures it is crucial to clearly label all the edges (relationships) and all the nodes (objects) with the use of color and text.

Additional criteria for graph visualization and layouts include minimizing edge crossings, maximizing graph symmetry, and ensuring consistent flow direction [17].

The proposed model for effective visualization of data structures in graph databases is based on two visualization layouts for structures of the database—topological layout and cluster layout.

A. Topological Layout of the Property Graph

In the topological layout of the property graph of the database, the data structures are organized in the form of the property graph presented in the example on the left side of Fig. 2. This form of visualization of a graph database is the most common and certainly the most natural way of visual presentation of a graph database, but is hard to use with databases containing a large number of objects (nodes of property graph) and relationships between these objects (edges of the graph).

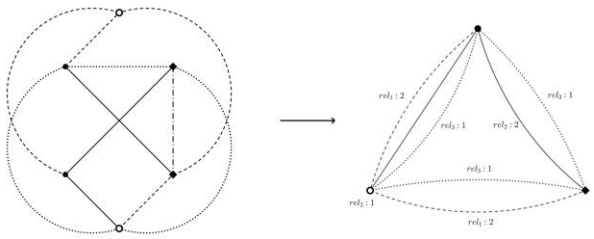

Fig. 2. Topological layout of graph database visualization (left) and clustered layout of graph database visualization (right).

Common and often unavoidable problems of such a visualization model include graph object crossings (either nodes or edges) and the clarity and readability of visualizations when working with large databases. In the case that objects of the database contain a large amount of *key:value* pairs, the visualization of such objects can not contain a listing of all the properties and their respective values. Therefore, the need for interactive development of individual nodes arises.

B. Clustered Layout of the Database

Clustering is the task of finding natural groupings in datasets [18]. When visualizing data structures in graph databases, there are natural groups present in the form of node and edge types in the database. In the clustered layout, we focus on the visualization of a graph database in the form of interconnected clusters. The right side of Fig. 2 shows such a visualization of a graph database consisting of three clusters labeled with the use of various shapes which are interconnected by three types of edges (marked by line type). Each interconnecting edge is denoted by relationship type label rel_n and the number of relationships of such type between two given clusters.

As can be noted in Fig. 2, the topological layout of the property graph contains four relationship types but in the clustered layout of the same database, only three types of relationships can be seen. This is caused by so-called inter-cluster relationships—relationships between two nodes of the same type. Such relationships constitute the need for the additional functionality of the proposed visualization model which could be called the developed view of clustered layout (Fig. 3).

This form of visualization allows a user to study the structure of individual clusters in the graph and, therefore, examine the relationships within clusters. The lower right cluster in Fig. 3 contains an inter-cluster relationship present in the original property graph of the database.

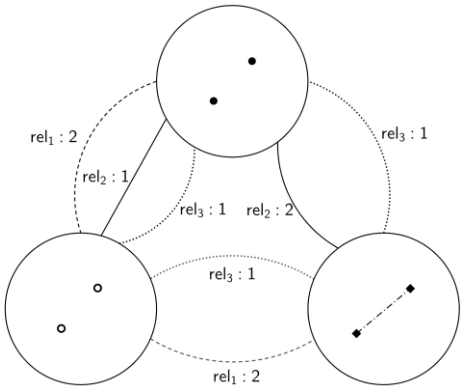

Fig. 3. Example of developed view of a clustered layout for visibility of inter-cluster relationships—in this case, the cluster of rhombic nodes contains a relationship between its elements.

IV. Case Studies of the Visualization on Model Graph Databases

The proposed visualization model, which includes both layouts presented in Section III of this work, was implemented using JavaScript, TypeScript, Cypher, and the Angular framework. In the following text, we describe the use of the proposed approach for the visualization of three model graph databases and compare the visualization with the native Neo4j tool.

A. Description of Databases Used in Case Studies

For the experimental evaluation of the proposed visualization model, the following three model graph databases were used:

- Messaging database. An original database comprising four discrete graph components, each containing 4 objects and 4 relationships, modeling a small social network.
- Library database. The second original database, structured into two graph components with the same number of objects as the Messaging database, but containing 32 relationships between these objects. This database models a simple library system for borrowing literature.
- Movie database. One of the two standard graph databases available in the Neo4j system, it models relationships between movies, actors, directors, and writers [19].

Each of these graph databases contains a different number of object and relationship types while also consisting of various instances for both. The number of objects, relationships, and object and relationship types for selected databases is presented in Table I, where a pair *label:number* denotes the label used as object/relationship title and number of instances of this object or relationship.

Fig. 4 presents a schematic visualization of data substructures in the model databases used in the experiments. These substructures are the only possible subgraphs contained in the property graph of the database (other than disconnected objects). This variety of size and structural composition of individual databases is the main reason for their inclusion in the presented case study of the proposed visualization model.

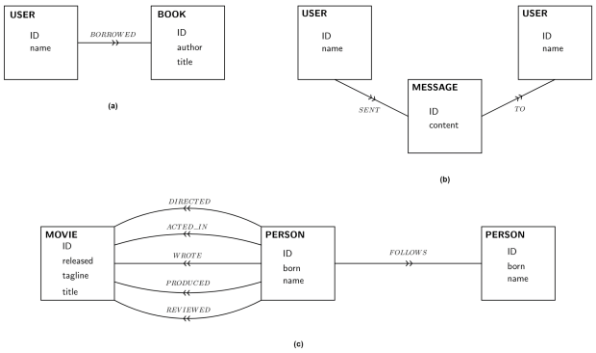

Fig. 4. Schematic visualization of data substructures in model databases used in experimental evaluation of the proposed visualization model-the subfigure, (a) presents objects in the Library database; subfigure, (b) contains the structure of objects in Messaging database; the subfigure, (c) contains objects in the graph database Movie.

TABLE I. Description of Model Databases

Database	Objects	Relationships
Messaging	User: 8 Message: 8	SENT: 8 TO: 8
Library	User: 8 Book: 8	BORROWED:32
Movie	Movie: 38 Person: 133	ACTED_IN: 172 DIRECTED: 44 PRODUCED: 15 WROTE: 10 FOLLOWS: 3 REVIEWED: 9

B. Visualization of Databases with the Use of the Proposed Model

As described in Section III, the core of the proposed visualization model is the visual representation of structures and substructures presented in an input graph database. Figs. 5–7 present the most basic form of visualization of the property graph of the database with the use of a topological layout. Specifically, Fig. 5 contains a visualization of the property graph for the Messaging database which consists of four discrete graph components, each composed of four objects and four relationships between these objects.

Fig. 6 shows the property graph of the Library database which similarly to the previous database—contains 16 nodes, but unlike the Messaging database, there are twice as many relationships between objects themselves. The property graph of this database consists of two components.

The largest of the databases used for the case study of the proposed visualization model is the Movie database consisting of more than 170 objects interconnected by more than 250 relationships. As seen in Fig. 7, the property graph of this database is composed of a single graph component.

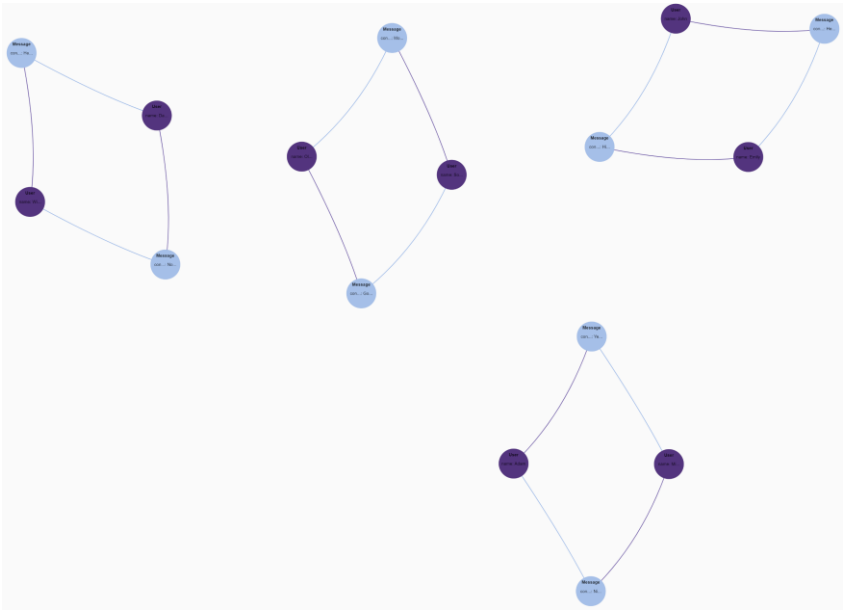

Fig. 5. Topological layout visualization of Messaging database—blue nodes describe objects of message type, purple nodes describe objects of user type; there are two types of relationships SENT (in blue) and TO (in purple).

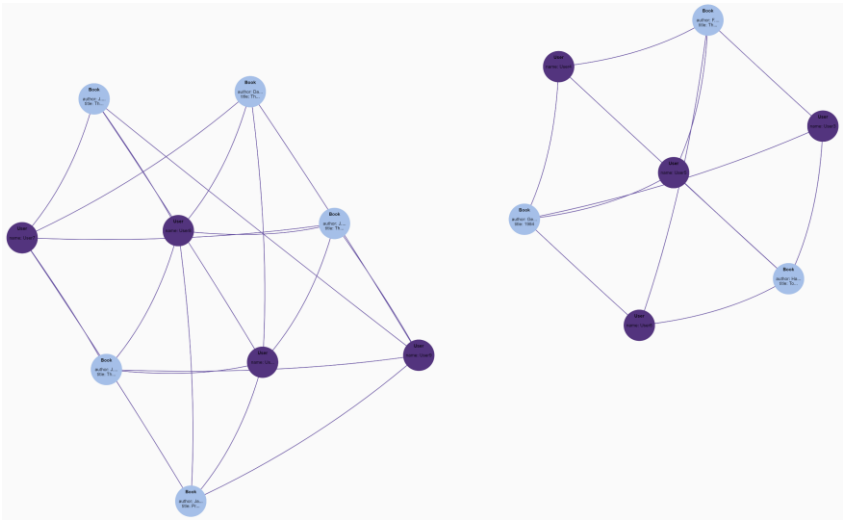

Fig. 6. Topological layout visualization of Library database—blue nodes describe objects of book type, purple nodes describe objects of user type.

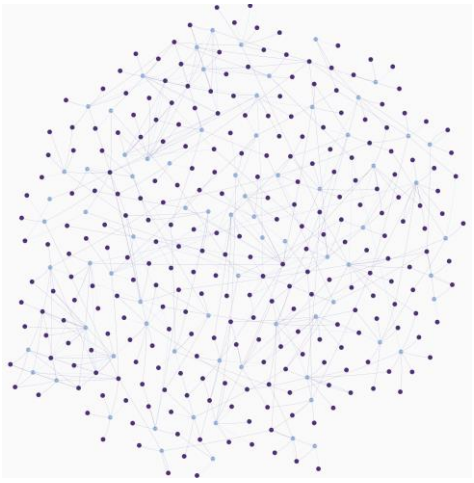

Fig. 7. Topological layout visualization of Movie database—blue nodes represent objects of *movie* type and dark purple nodes represent objects of person type.

Naturally, the visual representation of a large database needs the implementation of the above-mentioned interactivity elements. Figs. 8 and 9 focus on presenting specific properties of the proposed visualization model, namely:

- Scaling of the visualization is presented in Fig. 8, where the close-up of a pair of objects with their respective relationships is shown. Each object contains a label of its type (User or Book) and some of the *key:value* pairs of the object.
- Development of data objects is partially presented in both Figs. 8 and 9. When the user points to the edge between two objects, the type of relationship which is represented by this edge is visualized (e.g. BORROWED in Fig. 8). After selecting a specific node from the graph, it is developed and presented in the format depicted in Fig. 9.

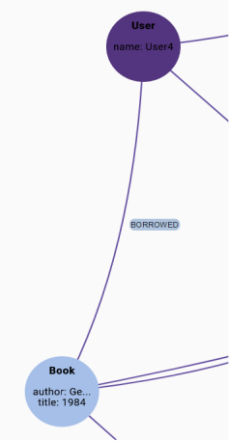

Fig. 8. Close-up of a pair of database objects with their respective relationships and interactive presentation of relationship label.

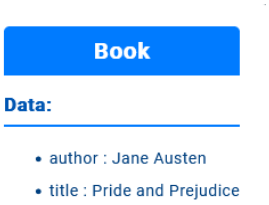

Fig. 9. Developed node of book type containing all *key:value* pairs of the object.

It is evident that Fig. 7 is difficult to read and navigate. This problem motivated the design and implementation of cluster layout for graph database visualization. Such layout for the Movie graph database is presented in Fig. 10 which contains two clusters—one for each object type and five relationships between these clusters. Other than the label of the relationship, the visualization adds the number of instances for each relationship.

Naturally, graph databases contain relationships between objects of the same type, and therefore, the need for closer inspection of individual clusters in clustered layout arises. Fig. 11 shows the development of part of the Person cluster from the Movie database. There are two types of nodes in this cluster—nodes without inter-cluster relationships and nodes with such relationships. The same options for interactivity from previous visualizations hold for a developed view of clusters.

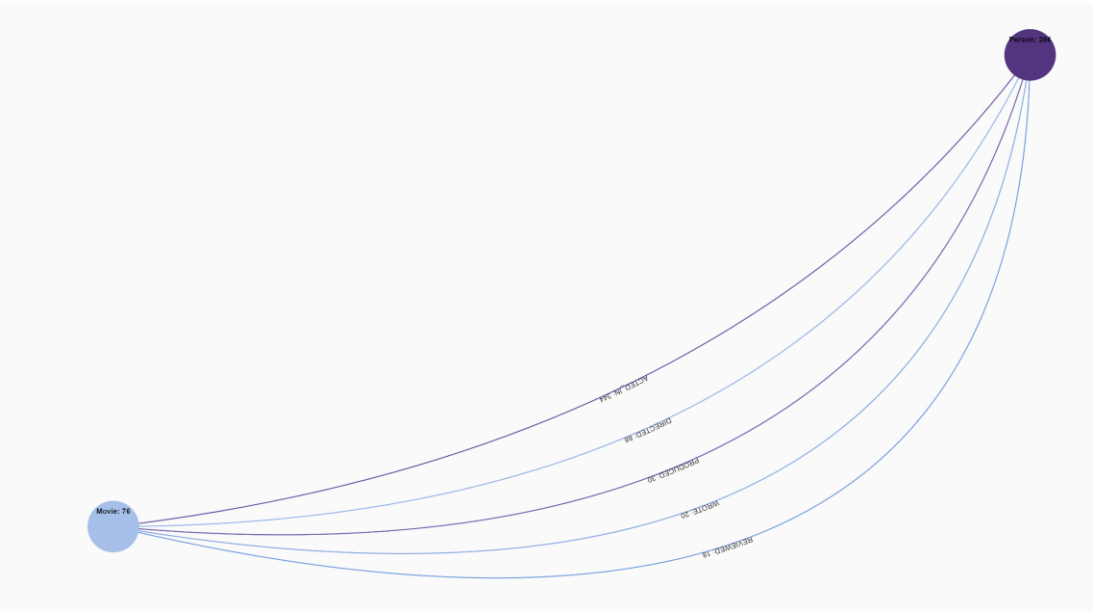

Fig. 10. Clustered layout visualization of Movie database—this layout presents condensation of the database into two types of objects (Movie, Person) and five types of relationships between these objects with the number of relationship instances.

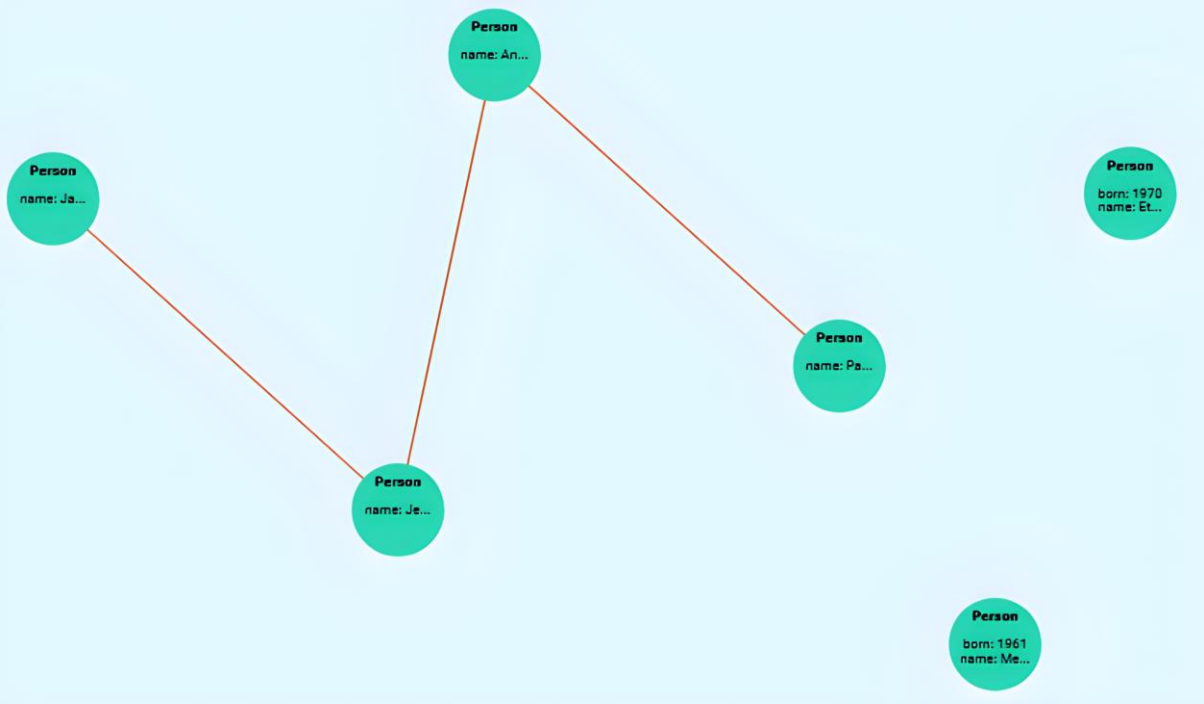

Fig. 11. Development of cluster Person in Movie database (partial view).

C. *Comparison of the Proposed Model to Native Neo4j Visualization*

The standard method of visualizing graph databases is the use of the native Neo4j visualization tool, the biggest advantage of which is its convenience, as it is implemented directly in the system. However, despite its integration into the Neo4j system, this tool has several significant shortcomings, including low readability for larger databases, limited interactivity, and constraints in displaying the number of vertices.

Fig. 12 presents (in some cases partial) visualization of graph databases used for experimental evaluation of the proposed approach. Fig. 12(a) shows a portion of the Movie database visualization, which, due to its size, posed significant challenges in the visual representation of databases in the tool:

- There is only one layout for property graph visualization in the Neo4j model—topological layout. This visualization has basic elements of interactivity, but there is no development of nodes possible in the tool.
- When visualizing larger databases, the tool has implemented a limit on the number of nodes, which can be displayed.

Fig. 12(b) and Fig. 12(c) present the visualization of Library and Message databases with the use of the native Neo4j model. Both visual representations of the databases, native Neo4j visualization and visualization via the presented model, are comparable in terms of handling crossing graph components. However, the proposed model excels in several areas where Neo4j visualization falls short, including scaling of visualization, object development, maximizing the ratio between color use and data, and visualizing graph databases as object type clusters.

The proposed graph database visualization model solves the issue of low readability and limited interactivity of large databases identified in Neo4j visualizer with the combination of:

- interactive scaling of the property graph,
- possibility of development of individual nodes,
- and for analytical purposes novel, a cluster visualization layout.

Similarly, the presented model does not implement any limit on the number of vertices (objects) present in the property graph of the database. This helps in the visualization of complete databases, which contain a large number of objects and relationships between these objects.

The most significant drawback of the tool is that the visualization is performed using separate complementary software. Consequently, users need to work with graph database software and visualization software separately.

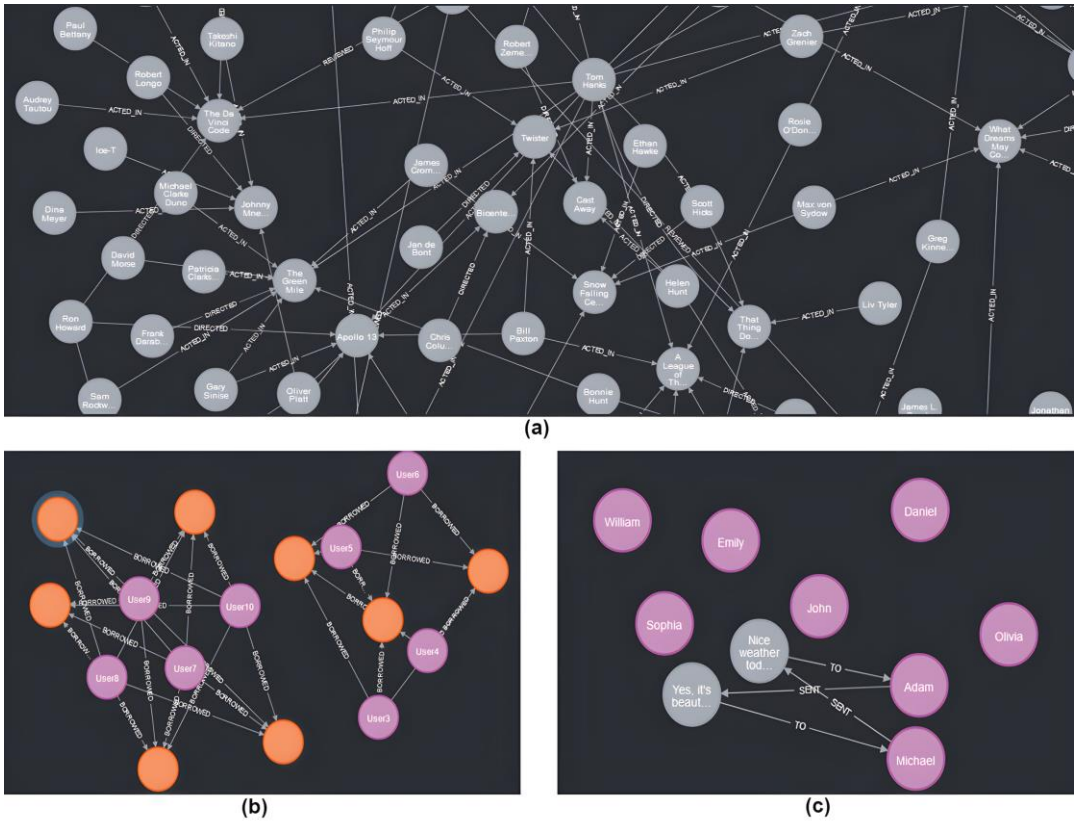

Fig. 12. Visualization of model databases with the use of native Neo4j tool: (a) Movie database, (b) Library database, and (c) Message database.

V. Conclusions

The main objective of this article was the design and implementation of a visualization model for graph databases with a focus on several effective visualization criteria, such as interactivity of visualization or graph object development. The novelty of presented research lies in proposing and implementing a novel graph object cluster visualization for graph databases, which enhances visual data analysis. Additionally, it introduces a novel tool focused on visualizing data structures within graph databases. The proposed model was subsequently used in the case studies on three graph databases of various sizes.

Although lacking the inherent advantage of native in-system integration typical in graph database visualization tools, the proposed approach successfully addresses challenges related to enhanced interactivity and effectiveness in visualizing graph databases.

There are several possible future work areas, which can be based on this research work. The most natural continuation of the work is a visualization of graph database queries in the property graph of the database. This task would involve computationally intensive graph searching and, therefore, possible use of parallel, distributed, or cloud computing.

On the other hand, there remains a significant opportunity in data analysis within graph databases. As humans are inherently visual creatures, visualizing data and its structures in these databases represents one of the most intuitive initial steps in understanding their content. Also, predictive analysis in the context of databases with a focus on the explainability of decisions through visualization is of interest for future research in the selected area.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Authors of the work contributed equally as follows: Adam Dudáš contributed to the conceptualization of visualization models, analysis, writing and revision of the manuscript. Adam Kleinedler contributed to software implementation, analysis, and writing of the manuscript. All authors had approved the final version of the manuscript.

Funding

The support of the Advtech_AirPollution project (Applying some advanced technologies in teaching and research, in relation to air pollution, 2021-1-RO01-KA220-HED-000030286) funded by the European Union within the framework of the Erasmus+ Program is gratefully acknowledged.

Code and Data Availability

The software tool and data created as a part of the presented research are freely available at the following link: *https://github.com/Eldam804/GraphDatabasViewer*

REFERENCES

[1] M. Besta, R. Gerstenberger, and E. Peter *et al.*, "Demystifying graph databases: Analysis and taxonomy of data organization," *System Designs, and Graph Queries. ACM Computing Surveys*, vol. 56, no. 2, pp. 1–4, 2023. doi: 10.1145/3604932

[2] S. Latif, Z. Mushtaq, and G. Rasool *et al.*, "Pragmatic evidence of cross-language link detection: A systematic literature review," *Journal of Systems and Software*, vol 206, 2023. doi: 10.1016/j.jss.2023.111825

[3] S. Timón-Reina, M. Rincón, R. Martínez-Tomás *et al.*, "An overview of graph databases and their applications in the biomedical domain," *Database*, 2021. doi: 10.1093/database/baab026

[4] M. Kvet, "Relational data index consolidation," in *Proc. Conference of Open Innovation Association, FRUCT*, 2021, pp. 215–221. doi: 10.23919/FRUCT50888.2021.9347614

[5] J. Janech, M. Tavač, and M. Kvet *et al.*, "Versioned database storage using unitemporal relational database," in *Proc. IEEE 15th International Scientific Conference on Informatics*, 2019, pp. 31–36. doi: 10.1109/Informatics47936.2019.9119269

[6] L. D. E. Pernas, A. Vichalkovski, and W. Steingartner *et al.*, "Automatic indexing for Mongodb," *New Trends in Database and Information Systems*, pp. 535–543, 2023.

[7] J. Dann, D. Ritter, H. Fröning, "Non-relational databases on FPGAs: Survey, design decisions, challenges," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–37, 2023. doi: 10.1145/3568990

[8] S. Ferilli, E. Bernasconi, and D. D. Pierro *et al.*, "A graph DB-based solution for semantic technologies in the future internet," *Future Internet*, vol 15, 2023. doi: 10.3390/fi15100345

[9] M. Hribernik, S. Tomažič, A. Umek *et al.*, "Unified platform for storing, retrieving, and analysing biomechanical applications data using graph database," *Journal of Big Data*, vol 10, no. 1, 2023. doi: 10.1186/s40537-023-00747-y

[10] D. Orlando, J. Ormachea, V. Soliani *et al.*, "TGV: A visualization tool for temporal property graph databases," *Information System Frontiers*, pp. 1–22, 2023. doi: 10.1007/s10796-023-10426-1

[11] Y. Wang and Y. Wang, "Research on network security visualization based on graph database," in *Proc. 2nd International Conference on Applied Mathematics, Modeling and Simulation (AMMS)*, 2018.

[12] W. Mueller, P. Idziaszek, K Przybył *et al.*, "Mapping and visualization of complex relational structures in the graph form using the Neo4j graph database," in *Proc. Eleventh International Conference on Digital Image Processing (ICDIP 2019)*, 2019. doi: 10.1117/12.2539707l

[13] N. Andrienko, G. Andrienko, G. Fuchs *et al.*, "Visual analytics for data scientists," *Springer Cham.*, 2020. doi: 10.1007/978-3-030-56146-8

[14] S. S. Skiena, *The Data Science Design Manual*, Springer Cham., 2017. doi: 978-3-319-55444-0

[15] E. R. Tufte, *The Visual Display of Quantitative Information*, Graphics Press, 1983. ISBN: 978-1930824133

[16] M. Udristioiu *et al.*, *Advanced Technologies of Big Data Processing and Analysis*, Academician Bookstore House, 2023. doi: 10.37609/akya.2633

[17] H. C. Purchase, "Metrics for graph drawing aesthetics," *Journal of Visual Languages & Computing*, 2002. doi: 10.1006/jvlc.2002.0232

[18] A. Michalíková, "Some notes on intuitionistic fuzzy equivalence relations and their use on real data," *Notes on Intuitionistic Fuzzy Sets*, vol. 28, no. 3, pp. 306–318. doi: 10.7546/nifs.2022.28.3.306-318

[19] Example datasets. Neo4j. (2024). [Online]. Available at: https://neo4j.com/docs/getting-started/appendix/example-data/

Copyright © 2024 by the authors. This is an open access article distributed under the Creative Commons Attribution License (CC BY-NC-ND 4.0), which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.