Optimizing Multi-Modal Transportation in Smart Cities: A Graph-Oriented Database Approach

Mihai Hulea
Automation Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
mihai.hulea@aut.utcluj.ro

Radu Miron
Automation Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
radu.miron@aut.utcluj.ro

Andrei Rusu
Innovation Office
NTT DATA Romania
Cluj-Napoca, Romania
rusu.andrei@nttdata.com

***Abstract*— This paper presents an innovative approach to enhancing urban transportation efficiency through the integration of multimodal transportation networks using graph databases, specifically OrientDB. Addressing the inefficiencies in traditional segregated passenger and freight movements, the study explores the potential of graph databases to effectively manage and optimize complex, interconnected urban transportation systems. Utilizing a practical implementation in Cluj-Napoca, Romania, the research demonstrates how this model can improve operational efficiency and resource allocation. The paper proposes a graph-oriented data model designed to store several types of transportation nodes and links effectively. Additionally, we explore a range of queries to demonstrate the practical usability and functionality of this data model.**

***Keywords*— Multimodal Transportation, Graph Databases, Urban Mobility**

I. Introduction

In the contemporary urban landscape, the efficient movement of both people and goods stands as one of the pillars of sustainable city living. Multimodal transportation, the use of various transport modes like buses, trains, bicycles, and walking in a single journey, is recognized as a vital component of urban mobility. The potential of multimodal transportation extends further when considering the integration of passenger and freight movements and it holds an important potential in streamlining city traffic. Currently, most urban transportation systems operate with a distinct separation between passenger and freight services. This segregation often leads to inefficiencies, such as underutilized capacity and overlapping routes, contributing to increased traffic congestion, pollution, and delays. The last mile of delivery is often the most expensive and least efficient part of the transportation journey. It can account for a significant portion of the total transportation costs due to factors like traffic congestion, parking limitations, and frequent stops. Cities often face challenges with congestion, especially in densely populated urban areas. The increase in delivery vehicles contributes to this congestion and leads to higher emissions. Integrating freight delivery with existing public transportation can lead to innovative solutions for last mile delivery challenges [9]. For instance, using public transit for parcel delivery during off-peak hours can optimize resource use. Another approach is proposed in [10] where a two-tiered solution is used for moving parcels into urban areas. Public transport routes are utilized for moving packages into urban areas, followed by delivery through smaller last-mile vehicles. A key feature is the use of automated parcel lockers at designated transit stops, allowing for asynchronous transfer of goods between transit and last-mile vehicles.

In the context of the described urban transportation landscape, the primary objective of this paper is to explore the application of graph databases in modeling a multi-modal transportation network considering several types of nodes related to both passenger and freight transportation. The proposal aims to explore the capabilities of graph databases to address the complexities and dynamic nature of modern urban transportation systems. Graph databases are ideal for modeling complex networks like transportation systems due to their inherent structure of nodes and edges, which naturally represent stations, junctions, routes, and connections. This research is partially sustained by the DELPHI project funded by the European Union under grant agreement No 101104263. The project aims to improve urban mobility in European cities by making it more efficient and sustainable by providing integrated freight and passenger transportation. Our study on using graph databases to optimize multimodal transportation directly supports DELPHI's goals.

II. State of the Art

Recently, numerous frameworks have emerged to handle graph-like data, reflecting the current importance of such data in various domains like social and biological networks. Graphs have long been employed to represent diverse domains, such as genetic interactions in science or user relationships in social networks. Recent work categorizes graph data management systems into two classes: graph databases and distributed graph processing structures [1].

Graph databases store data by organizing it into nodes, edges, and relationships, eliminating the need for indexes (index-free adjacency). Data is conceptually organized in nodes, each linked to another through edges, forming relationships. This structure allows direct lookups between connected nodes, eliminating the need for indexing. Modern social networking sites like Facebook utilize graph databases to efficiently store large amounts of data [2].

Graph databases represent one of the four categories of NoSQL databases, alongside key-value, document, and column-oriented databases. NoSQL database management systems emerged in response to the limitations of traditional technologies and the challenges of the Web 2.0 era, and their introduction can be traced back to Google's MapReduce (2004) and BigTable (2006) [3].

Since the 1980s, the relational model has dominated the computer industry for data storage and retrieval. However, its decline is evident due to a rigid schema, hindering the addition of new relationships. Another drawback is the inefficiency in handling the growing volume of data, as joining numerous tables becomes cumbersome [4]. The relational model was introduced by E.F. Codd in 1970, to overcome the problems of the network and hierarchical models database systems [5]. Although accepted by the academic community, the adoption of the relational databases started with the introduction of IBM’s System R [6].

Relational databases have a fixed structure, known as a schema, established at database creation. All inserted data must adhere to this schema, emphasizing the importance of proper database design from the start. Anticipating future data requirements is crucial for accommodating additional needs over time. Primary and foreign keys are used to maintain referential integrity, ensuring foreign key values match primary key values in parent tables. However, interpreting foreign key values without a query connecting tables poses challenges, as these values are often automatically generated numbers [2].

A graph database employs vertices (nodes) and edges (relationships) to store data. The properties are the attributes of the nodes. In contrast to the relational model, where for example each person and their associations may be stored in separate tables, a graph database represents individuals as nodes and their connections as edges. Graph databases excel in graph operations such as traversals and shortest path calculations. Given the abundance of accessible data, individuals may find value in exploring graph representations or adopting graph databases for their scalability and querying capabilities. Graph databases are particularly effective in visualizing data with numerous many-to-many relationships [7].

Reference [1] presents a comparison between the most popular implementations of graph databases. The list of the considered databases includes: Neo4j, DEX, Infinite Graph, Infogrid, HyperGraphDB, Trinity, and Titan.

Graph-oriented databases are increasingly recognized as a natural fit for addressing challenges such as storing and managing information related to transportation networks. In the literature, there are initiatives that highlight the use of these databases to represent traffic networks effectively. Additionally, various algorithms have been employed to optimize traffic routing. These initiatives demonstrate the potential of graph-oriented databases in enhancing the efficiency and effectiveness of transportation network management. In [11] and [12] a solution for representing multimodal transportation network using Neo4J is proposed. In addition, [12] proposes a solution for configuring and saving graph routes using a mobile application and GPS data. [13] presents a framework for a multimodal graph database that encompasses both public and private transit modes, such as driving, walking, cycling, bus, and rail. Based on detailed investigation and tests performed in a London use case it is concluded that a graph database is more suitable for representing and managing multimodal transportation networks compared with traditional relation database.

III. Proposed Solution

The proposed solution was implemented based on OrientDB which is an open source, multi-model database that supports graph, document, key-value, and object-oriented database models. In choosing OrientDB key advantages include its hybrid model, allowing for a blend of graph, document, object, and key/value models in a single application. Its flexible schema supports both strict and evolving data models. OrientDB's SQL-like query language, grounded in familiar SQL syntax, enhances ease of use. Additionally, its open-source nature under the Apache 2 license presents a cost-effective solution.

A graph is a representation of a network-like structure, comprising Vertices (also referred to as Nodes) connected by Edges (also known as Arcs). OrientDB's graph model adheres to the property graph concept, outlining the following key components:

- Vertex: An entity capable of linking with other Vertices, featuring the following essential properties: unique identifier, the set of incoming Edges, the set of outgoing Edges.
- Edge: An entity linking two Vertices, possessing the following essential properties: unique identifier, link to an incoming Vertex, link to an outgoing Vertex, label defining the type of connection/relationship between the head and tail vertices.
- Apart from these mandatory properties, both vertices and edges can accommodate a set of custom properties defined by users. This flexibility allows vertices and edges to resemble documents, providing additional customization.

Table I presents a comparison between the relational model and the OrientDB graph model.

TABLE I. RELATIONAL AND ORIENTDB GRAPH MODELS

Relational Model	**OrientDB Graph Model**
Table	Class that extends “V” (for Vertex) and “E” (for Edge)
Row	Vertex
Column	Property of a Vertex or an Edge
Relationship	Edge

In conjunction with the graph model, OrientDB supports the object model, thus enabling inheritance and polymorphism. These are powerful features for designing the data model and for querying the database [8].

A. *Graph Oriented Data Model for Multi-Modal Transportation*

Figure 1 presents the object-oriented data model for the proposed graph. The classes presented in the data model are split between vertices (or nodes) and edges. To define vertex and edge classes in OrientDB the V and E predefined classes must be extended, respectively.

As mentioned earlier the data model is split between nodes and edges. The model defines a base node class, *multimodal_node,* that contains the common attributes of all the other node types in the graph: *id*, *name*, *description,* and the geographical coordinates (*latitude* and *longitude*). *Multimodal_node* is further extended by concrete nodes: *bus_station*, *bike_station*, *closed_parking*, *cargo_warehouse* and *parcel_locker.* The classes *bike_station*, *parcel_locker* and *closed_parking,* apart from the inherited attributes, have two other attributes: *capacity* and *availability.*

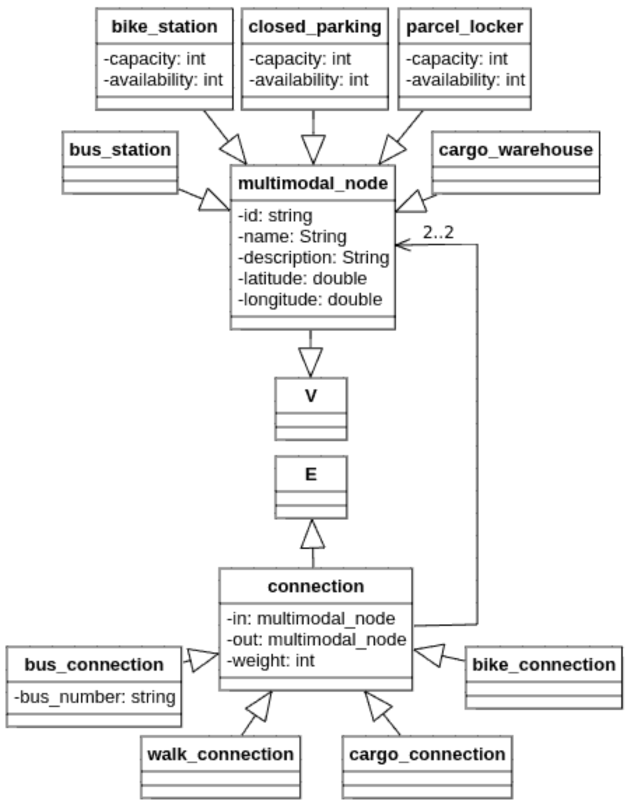


Fig 1. Object oriented data model

Regarding the edges, the model contains a base, namely *connection*. The concrete edge types are: *bus_connection, walk_connection, cargo_connection and bike_connection.* These classes inherit the following attributes from the parent *connection: in, out* and *weight.* The *in* and *out* attributes have the purpose of storing the references to the input and output nodes that the edge connect, respectively. The *weight* attribute represents the effort needed to get from the input node to the output node connected by the current edge. While the weights of the bus and cargo connections are constant (=1), the weights of the bike and walk connections are proportional with the distance between the two nodes.

B. *Graph Network*

As a proof of concept, the current paper proposes the graph of a *partial public transport network* from Cluj-Napoca, Romania. For populating the graph, real data from several public sources was used:

- bus stops: *https://api.tranzy.dev/v1/opendata/stops*
- bike stations: *https://portal.clujbike.eu/stations.txt*
- closed parking: *https://data.e-primariaclujnapoca.ro/sitpark.json*
- parcel locker & other data: *Google Maps* and *Open Street Map*

Figure 2 presents a partial image of the graph implementation. The full graph is available online at:

https://github.com/dcs-research/paper-multimodal-graph-db-aqtr-2024/blob/main/graph.png

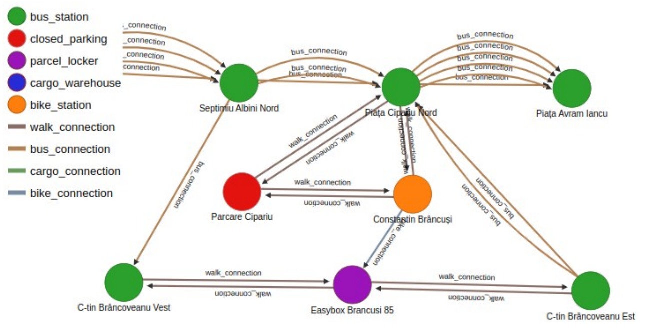


Fig 2. Partial public transport graph

C. *Queries for Finding Optimal Routes*

The model of the public transport network represented as a graph can be particularly useful for finding the shortest path between two nodes. For example, one use case would be the use of the public transportation system for package delivery from the warehouse to a specific parcel locker. Figure 3 presents the OrientDB query, and the result represented as a sub-graph.

```
SELECT expand(path) FROM (
  SELECT dijkstra($from, $to, 'weight') AS path
  LET
    $from = (SELECT FROM multimodal_node WHERE name='FAN Courier Warehouse'),
    $to = (SELECT FROM multimodal_node WHERE name='Easybox Brancusi 85')
  UNWIND path
)
```

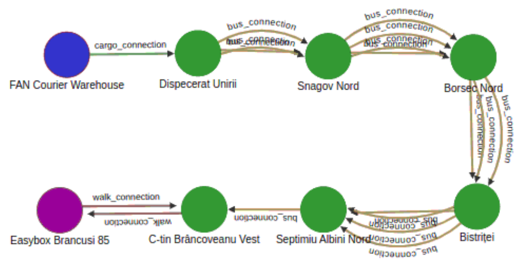
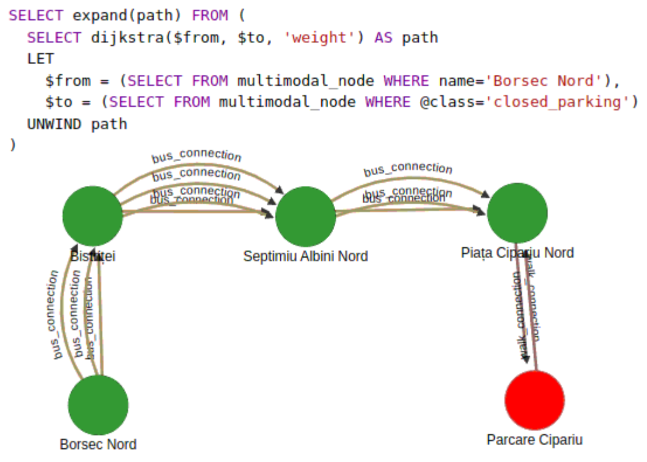


Fig 3. Shortest path between the warehouse and parcel locker

D. *Queries for Finding Path to Closest Node Type*

Another common use case is to find the path to the closest node type. For example, finding the path from a bus stop to the closest closed parking (see Figure 4).

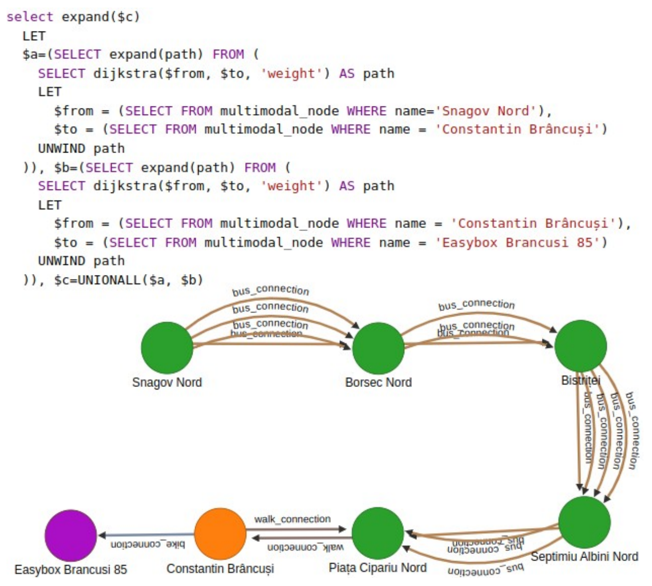


```
SELECT expand(path) FROM (
  SELECT dijkstra($from, $to, 'weight') AS path
  LET
    $from = (SELECT FROM multimodal_node WHERE name='Borsec Nord'),
    $to = (SELECT FROM multimodal_node WHERE @class='closed_parking')
  UNWIND path
)
```

Fig 4. Shortest path to the closest closed parking

E. Queries for Finding Optimal Route with Constraints

OrientDB facilitates the execution of queries enabling the identification of the shortest path between two nodes while passing through an intermediate node. To illustrate this capability, consider a last-mile package delivery solution that leverages both public buses and the city's free bike-sharing service. In this scenario, the delivery agent can be efficiently routed from a starting bus stop (Snagov Nord) to the parcel locker containing the package (Easybox Brancusi 85). This route is optimized by passing through the nearest bike station in proximity to the designated parcel locker (see Fig. 5).



```
select expand($c)
  LET
  $a=(SELECT expand(path) FROM (
    SELECT dijkstra($from, $to, 'weight') AS path
    LET
      $from = (SELECT FROM multimodal_node WHERE name='Snagov Nord'),
      $to = (SELECT FROM multimodal_node WHERE name = 'Constantin Brâncuși')
    UNWIND path
  )), $b=(SELECT expand(path) FROM (
    SELECT dijkstra($from, $to, 'weight') AS path
    LET
      $from = (SELECT FROM multimodal_node WHERE name = 'Constantin Brâncuși'),
      $to = (SELECT FROM multimodal_node WHERE name = 'Easybox Brancusi 85')
    UNWIND path
  )), $c=UNIONALL($a, $b)
```

Fig 5. Last mile delivery solution

IV. Conclusions

The present paper highlights the role of multimodal transportation in enhancing urban mobility. By integrating passenger and freight movements through a multi-modal network modeled in a graph database, cities can significantly improve the efficiency of their transportation systems. This integration promises to reduce traffic congestion, lower pollution, and streamline the last-mile delivery process. The use of graph databases, as demonstrated in the OrientDB model, allows for a more dynamic and flexible representation of transportation networks, accommodating the complex and ever-changing nature of urban transport systems. The adoption of graph databases offers a robust solution for managing the complexities inherent in multimodal transportation networks. This approach has advantage of relational database models by providing more intuitive and efficient methods for data representation and querying, especially in systems characterized by many-to-many relationships and network-like structures. The practical application of the proposed graph database model in the public transport network of Cluj-Napoca highlights the potential of this approach in real-world scenarios.

Acknowledgment

Part of this research has been conducted as part of the DELPHI project, which is funded by the European Union, under grant agreement No 101104263. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them. In addition, this research was collaboratively conducted by - Technical University of Cluj-Napoca, Romania.

References

[1] D. S. Rawat and N. K. Kashyap, "Graph Database: A Complete GDBMS Survey," International Journal for Innovative Research in Science & Technology, vol. 3, no. 12, May 2017.

[2] S. Paul, A. Mitra, and C. Koner, "A Review on Graph Database and its representation," in 2019 International Conference on Recent Advances in Energy-efficient Computing and Communication (ICRAECC), Mar. 2019, pp. 1–5.

[3] M. Indrawan-Santiago, "Database Research: Are We at a Crossroad? Reflection on NoSQL," in 2012 15th International Conference on Network-Based Information Systems, Melbourne, Australia: IEEE, Sep. 2012, pp. 45–51.

[4] S. Batra and C. Tyagi, "Comparative Analysis of Relational and Graph Databases," International Journal of Soft Computing and Engineering (IJSCE), vol. 2, no. 2, 2012.

[5] E. F. Codd, "A relational model of data for large shared data banks," Commun. ACM, vol. 13, no. 6, pp. 377–387, Jun. 1970.

[6] M. M. Astrahan et al., "System R: relational approach to database management," ACM Trans. Database Syst., vol. 1, no. 2, pp. 97–137, Jun. 1976.

[7] N. Tyagi and N. Singh, "Graph Database-An Overview of its Applications and its Types," International Journal of Computer science engineering Techniques., vol. 2, no. 3, 2017.

[8] "Data Modeling · OrientDB Manual." Accessed: Jan. 25, 2024. [Online]. Available: https://orientdb.org/docs/3.0.x/datamodeling/Tutorial-Document-and-graph-model.html

[9] D. Delle Donne, L. Alfandari, C. Archetti, and I. Ljubić, "Freight-on-Transit for urban last-mile deliveries: A strategic planning approach," Transportation Research Part B: Methodological, vol. 169, pp. 53–81, Mar. 2023, doi: 10.1016/j.trb.2023.01.004.

[10] I. Azcuy, N. Agatz, and R. Giesen, "Designing integrated urban delivery systems using public transport," Transportation Research Part E: Logistics and Transportation Review, vol. 156, p. 102525, Dec. 2021, doi: 10.1016/j.tre.2021.102525.

[11] P. Wirawan, D. Riyanto, D. Nugraheni, and Y. Yasmin, "Graph Database Schema for Multimodal Transportation in Semarang," Journal of Information Systems Engineering and Business Intelligence, vol. 5, p. 163, Oct. 2019, doi: 10.20473/jisebi.5.2.163-170.

[12] B. Vela, J. M. Cavero, P. Cáceres, A. Sierra, and C. E. Cuesta, "Using a NoSQL graph oriented database to store accessible transport routes," in Proceedings of the Workshops of the EDBT/ICDT 2018 Joint Conference, 2018, pp. 62–66.

[13] S. Park and T. Cheng, "Framework for constructing multimodal transport networks and routing using a graph database: A case study in London," Transactions in GIS, vol. 27, no. 5, pp. 1391–1417, 2023, doi: 10.1111/tgis.13071.