A Model and Query Language for Multi-modal Hybrid Query

Chuan Hu
huchuan@cnic.cn
CNIC, Chinese Academy of Science
University of Chinese Academy of Sciences
Beijing, China

Zihao Zhao*
airzihao@gmail.com
Huawei Technology Company
Beijing, China

Along Mao
almao@cnic.cn
CNIC, Chinese Academy of Science
University of Chinese Academy of Sciences
Beijing, China

Zhihong Shen†
bluejoe@cnic.cn
CNIC, Chinese Academy of Science
Beijing, China

ABSTRACT

As data grows exponentially, its diversity also increases, including both structured forms and unstructured forms like audio, images, and videos. Advances in AI have improved our ability to analyze unstructured data, leading to the use of multimodal hybrid queries that blend structured and unstructured data. However, database systems struggle due to the lack of adequate data models for multimodal data and languages for these hybrid queries. This paper extends the property graph model to represent multimodal data and their semantic information, introducing essential functions for hybrid graph queries. A high-level graph query language, Cypher-Plus, is presented, capable of expressing hybrid queries like "Give me the friends of the friends of Mary, who have blond hair and are younger than 30 years old." A Neo4j-based implementation and experiments over synthetic and real-world datasets demonstrate the approach's plausibility.

CCS CONCEPTS

• **Information systems** → **Query languages**; **Database design and models**.

KEYWORDS

Data Model, Query Languages, Multi-modal Data, Graph Databases

ACM Reference Format:
Chuan Hu, Zihao Zhao, Along Mao, and Zhihong Shen. 2024. A Model and Query Language for Multi-modal Hybrid Query. In *36th International Conference on Scientific and Statistical Database Management (SSDBM 2024), July 10–12, 2024, Rennes, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3676288.3676291

*Zihao Zhao finished his work at Computer Network Information Center, Chinese Academy of Science.
†Zhihong Shen is the corresponding author of this paper.

SSDBM 2024, July 10–12, 2024, Rennes, France

ACM ISBN 979-8-4007-1020-9/24/07.
https://doi.org/10.1145/3676288.3676291

1 INTRODUCTION

Real-world data is inherently multimodal, reflecting a variety of different data types such as text, images, and videos. In contemporary applications, these multimodal data are often interrelated, fused together in unstructured and structured data formats. Significant advances in artificial intelligence have greatly enhanced our ability to derive insights from multimodal data. Algorithms powered by AI, including natural language processing, computer vision, and speech recognition, unlock the latent potential of multimodal data. This transformative capability paves the way for multimodal hybrid queries.

Multimodal hybrid query. A hybrid query [9] refers to a type of query that combines both structured and unstructured data in a single query. In other words, it is a query that allows users to search for information by considering not only specific attributes or properties of data (structured data) but also the content or features within the data itself (unstructured data). Hybrid queries are used when there is a need to jointly search and analyze both structured and unstructured data for various purposes.

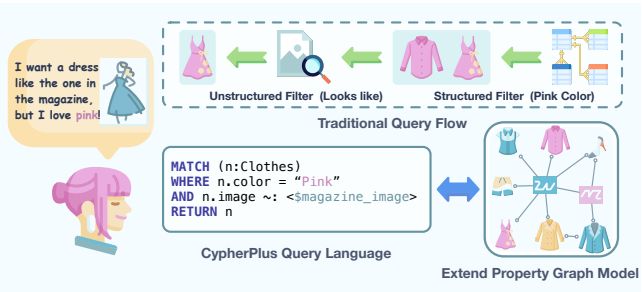

Figure 1: An Example of Multimodal Hybrid Query.

Example 1.1 (Multimodel Hybird Query [9]). As shown in Figure 1, Alice wants to buy a dress that is the same length as the picture, but she prefers pink. This query comprises information from two modalities: an image and text. The term "*Pink*" serves as a structured filter condition, while the similarity to the query image constitutes an unstructured filter condition. Multimodal hybrid queries such as these have enhanced expressiveness and higher accuracy, bringing significant value to emerging applications.

Challenges in database support for multimodal data. Yet, despite the growing importance of multimodal data analysis, the database field lags behind in providing robust support for such data types. Common practices involve storing unstructured multimodal data in file or object systems and establishing associations with structured data through intricate paths. This approach, however, falls short of efficiency and raises challenges in data management and consistency. The root of this challenge can be traced to two fundamental issues: first, existing data models struggle to encapsulate the complexities of multimodal data adequately; second, there is a notable absence of query languages adept at articulating multimodal hybrid queries. As a result, a critical gap remains in the database landscape.

Our method. In this paper, we propose a solution. Building upon the foundation of the property graph model [1], we have designed an extended property graph model capable of representing both structured and unstructured data simultaneously, directly describing multimodal information. We have further extended the graph query language, Cypher [4], to create CypherPlus, a multimodal graph query language.

Why use graph? Multimodal data inherently exhibits intricate interrelationships, and graph models excel in expressing these connections. Graphs offer the flexibility required to depict complex associations found in the natural world, enabling more agile and expressive data representation. Graphs have the unique advantage of integrating comprehensive information related to an entity into a single node while articulating associations through interconnected edges [6, 7]. This feature makes data representation more natural and intuitive [2]. As a result, in many real-world applications, data is organized and managed in the form of graphs. Prominent examples include social networks [3] and the implementation of smart city infrastructures [8].

In order to enable the property graph model and graph query languages to support multimodal data, several challenges need to be addressed. These challenges include: (1). The traditional graph models fail to represent the semantic information of multimodal data and further carry out the query operations on them. (2). Potential features in multimodal data are too many [5] to be enumerated. Existing graph query languages do not provide a mechanism to dynamically define, extract or compare semantic features. (3). The query language should have semantics to represent the essential operations to query multimodal data on the graph.

In summary, we make the following contributions:

- **Data Model:** We design extended property graph model, which supports the representation of semantic information of multimodal data on graphs. We describe the basic operations required for hybrid query.
- **Query Language:** We design CypherPlus, a user-friendly query language that provides the semantics needed to query multimodal data on graphs.
- **Prototype System:** We built a prototype database system based on Neo4j to demonstrate the advantages of the proposed model and query language.

We present the above contributions in turn in sections 2, 3, and 4, respectively. Experimental evaluation is presented in Section 5.

2 EXTENDED PROPERTY GRAPH MODEL

2.1 Data Structure

An extended property graph is a directed labeled multigraph with the special characteristic that each node or edge could maintain a set (possibly empty) of property-value pairs. The values could be unstructured data items, and each unstructured property could have a set of subordinate property-value pairs.

In applications, both structured and unstructured data describe the details of objects and relationships, i.e., the properties of nodes and relationships. Objects (e.g., Person) are modeled as nodes, and relationships (e.g., buy) are treated as edges. Unstructured data can contain multiple kinds of information, such as features and textual content. This information is called **semantic information** and can be regarded as properties of unstructured data, helping to further understand the data. The extended property graph treats unstructured data items as properties of nodes/relationships, referring to the semantic information as **sub-properties**. An unstructured property can have multiple sub-properties, which can be seen as a mutable map where keys are the names of sub-properties and values are semantic information. The size of the map can be extended as needed. We give the definition of sub-property as follows:

Definition 2.1 (Semantic information). Semantic information is the meaning of unstructured data content.

Definition 2.2 (Unstructured property). The unstructured data are regarded as the properties of nodes/edges, which are called unstructured properties. A node/relationship could have multiple unstructured properties.

Definition 2.3 (Sub-property). Sub-properties are the semantic information in unstructured property:

$$< up > \longrightarrow subPropertyKey =< semantic\ information > \quad (1)$$

In the extended property graph model, only nodes and relationships are identified, the unstructured data items and semantic information do not have unique identifiers. Users can specify nodes/relationships and their properties to obtain the unstructured properties. Further, one can access a specific sub-property item, by finding the unstructured property first.

2.2 Operations of Querying Unstructured Data

For the acquisition of semantic information of unstructured property, we introduce the semantic information extraction function ϕ.

Definition 2.4 (Semantic information extraction function ϕ). A finite partial function that extract specified semantic information from unstructured data as follows:

$$si = \phi(ud, sn) \quad (2)$$

In this definition, the si is semantic information, ud is an unstructured data and sn is a semantic name. For example, the query image q in the Figure 2 can likewise extract its semantic information: $\phi(q,$ color) = black.

The semantic information extraction function can also be used as sub-property extraction function:

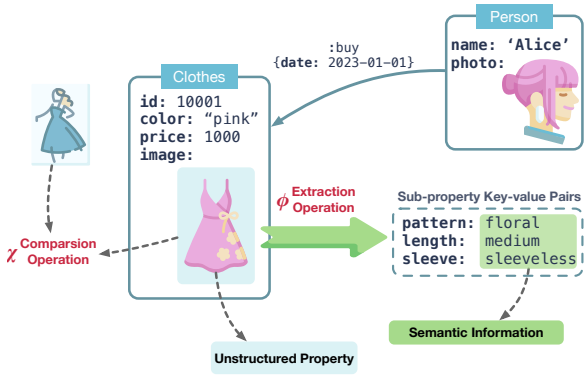

Figure 2: Extended Property Graph Model

Definition 2.5 (Sub-property extraction function ϕ). A finite partial function that maps a sub-property key to a sub-property value (semantic information) as follows:

$$\phi : \mathcal{V} \times \mathcal{SP} \rightarrow SET^{+}(\mathcal{SV}) \tag{3}$$

In this definition, $\mathcal{V}$ is the property value and $\mathcal{SP}$ is the set of sub-property names. $SET^{+}(\mathcal{SV})$ is the set of sub-property values.

Consider the node n_1 in Figure 2, the *price* and the *image* are the properties, and the *color* and *sleeve* are the sub-property. The sub-property extraction of the node(denoted by n) could be expressed in the following ways:

- $\phi(n.\text{image, color})$ = black
- $\phi(n.\text{image, sleeve})$ = short

We give a definition of comparison between unstructured data as follows:

Definition 2.6 (Comparison operation χ). The comparison operation χ is to compare two unstructured data by the semantic information:

$$\chi(\phi(d_1, sn_1), \phi(d_2, sn_2), \iota) =< compare\ result > \tag{4}$$

In this equation, d_1 and d_2 represent a unstructured data item, respectively, sn_1 and sn_2 mean two semantic name, and ι is the comparison rule between semantic information. For example, in Figure 2, the image of dress n is similar to the query image q, then the comparison operation and its result would be like:

$$\chi(\phi(n.photo, feature), \phi(q, feature), SEMIN) = TRUE \tag{5}$$

Note the *SEMIN* in the equation is a comparison rule, it means to judge whether a semantic information item is similarly included in another one.

3 CYPHERPLUS SYNTAX AND SEMANTICS

This section begins with an introduction to the syntax of CypherPlus, detailing the extensions we have made to Cypher.

3.1 Semantics of CypherPlus

The extended property graph model introduces new operations about querying unstructured data on the graph, including creating unstructured data items, extracting the semantic information and computing with them. We develop CypherPlus to facilitate these operations with newly proposed semantics:

Table 1: Logical Comparison Operators of Unstructured Data

Symbol	Description	Example
::	The similarity between x and y.	x::y = 0.7
~:	Is x similar to y.	x~:y = true
! :	Is x not similar to y.	x! :y = false
<:	Is x contained in y.	x<:y = true
>:	Is y contained in x.	x>:y = false

- **Unstructured Property Literal** is used to represent the literal value of unstructured property, employing a format delineated as <schema://path>. Here, the schema component can encompass various protocols such as FILE, HTTP(S), FTP(S), BASE64, and others.
- **Sub-property Extraction Operator** (represented as ->) is the semantic symbol of semantic information extraction function. It connects a unstructured property and a semantic name, meaning to obtain specific semantic information value from the data item, by invoking the semantic information extraction function.
- **Semantic Comparison Operators** provide a series of opertor, as shown in Table 1, to support logical relationship comparisons between specific semantic information. For example, ~: can be used to compute the similarity between two unstructured properties.

3.2 Query Statement Examples

Query 3.1. *Query the database for all women who have blond hair and are younger than 30 years old.*

```
MATCH (p:Person)
WHERE p.photo->hair_color = 'blond'
    AND p.age < 30 AND p.gender = 'female'
RETURN p
```

For more query examples, see the technical report of the open source project[1].

4 IMPLEMENTATION

Query Parser Implementation. OpenCypher[2] is an open-source initiative derived from Cypher, the prominent query language associated with graph databases. Building upon its foundational infrastructure, we have integrated the contents of CypherPlus[1], encompassing additions such as the type system, expressions, and functions. It now includes the BLOB (Binary Large Object) data type, enhancing the storage and retrieval of unstructured data like images and videos. Accompanying BLOB functions, such as Blob.fromBytes() and Blob.toBytes(), facilitate data management tasks. Additionally, new expression capabilities include Unstructured Property Literals, Sub-property Extraction Operators, and

[1]https://github.com/grapheco/cypherplus-front-end
[2]http://opencypher.org

Semantic Comparison Operators, supported by customized execution processes and flexible AI algorithm integration for tasks like image analysis.

Prototype Database System. PandaDB [10] is a prototype database system built on Neo4j, now open-sourced[3] and successfully applied in various projects. Its architecture includes a Neo4j-based storage engine, an BLOB store for large binary objects, and a query and execution engine for CypherPlus queries, integrating AI services for unstructured data handling. The AI service extracts semantic information from unstructured data using customizable AI models.

5 EXPERIMENTS

This section evaluates the performance of CypherPlus in parsing and generating query plans based on some representative query statements.

Setup. This experiment uses Cypher as the baseline and compares the time consuming parsing of Cypher and CypherPlus, so as to evaluate the impact of the newly added syntactic features in CypherPlus on parsing and generating query plans.

The experiment utilizes test statements derived from LDBC-SNB [3], a standardized benchmark for evaluating the efficiency of graph database management systems in managing social network data. For this experiment, we use two sets of query statements, where for Cypher queries we directly use the query statements for short-tasks **(S1-S7)** and update-tasks**(U1-U8)** in LDBC-SNB. For CypherPlus, we extend Cypher by adding the literals and operators described in Section 3.

The baseline Cypher compiler adopts the OpenCypher Frontend v9.0. The runtime environment comprises Scala 2.12.8 and Java 1.8, operating on MacOS 14.4 with an Apple M1 Pro processor and 16GB RAM.

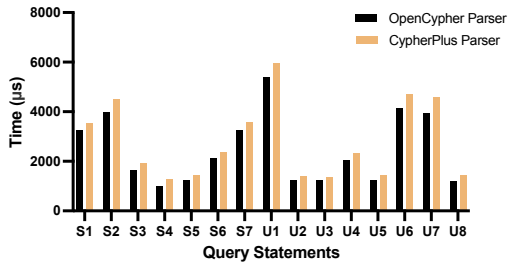

Figure 3: Parsing time comparison experiment.

Experiment 1. In the first experiment, we executed Cypher statements for each of the 15 tasks in the LDBC-SNB benchmark, as well as the corresponding CypherPlus statements with extensions. The experimental results are presented in Figure 3. The data indicates that, on average, CypherPlus is marginally slower than Cypher by 3% to 9% during the parsing process. Considering the minor extensions implemented in CypherPlus, this result is deemed acceptable. Consequently, it can be concluded that the extensions introduced in CypherPlus have minimal impact on parsing and query plan generation.

[3]https://github.com/grapheco/pandadb-v0.1

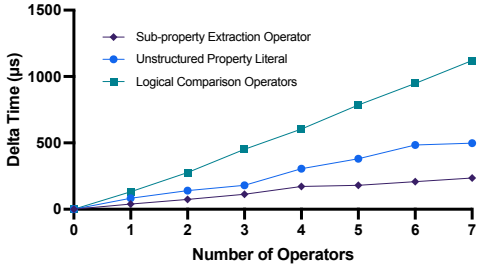

Figure 4: Impact of different number of new operators in query statements on parsing time.

Experiment 2. In the second experiment, we evaluated the effect of different numbers of new operators on query language parsing time. We conducted experiments separately for the three categories of operators mentioned in Section 3, and the results are depicted in Figure 4. Here, the X-axis represents the number of different new operators ($i = 0 - 7$) and the Y-axis represents the additional parsing time ($t_i - t_0$). It is evident that the additional time overhead of the three categories of operators exhibits an approximately linear relationship with the number of operators, indicating that in CypherPlus, the utilization of additional new operators does not result in disproportionately increased overhead.

ACKNOWLEDGEMENT

This work was supported by the National Key R&D Program of China(Grant No.2021YFF0704200) and Informatization Plan of Chinese Academy of Sciences(Grant No.CAS-WX2022GC-02)

REFERENCES

[1] Renzo Angles. 2018. The Property Graph Database Model.. In *AMW*.
[2] Renzo Angles and Claudio Gutierrez. 2008. Survey of graph database models. *ACM Computing Surveys (CSUR)* 40, 1 (2008), 1–39.
[3] Orri Erling, Alex Averbuch, Josep Larriba-Pey, Hassan Chafi, Andrey Gubichev, Arnau Prat, Minh-Duc Pham, and Peter Boncz. 2015. The LDBC social network benchmark: Interactive workload. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 619–630.
[4] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. 2018. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 international conference on management of data*. 1433–1445.
[5] Pui Yi Lee, Wei Ping Loh, and Jeng Feng Chin. 2017. Feature selection in multimedia: the state-of-the-art review. *Image and vision computing* 67 (2017), 29–42.
[6] Jan Paredaens, Peter Peelman, and Letizia Tanca. 1995. G-Log: A graph-based query language. *IEEE Transactions on Knowledge and Data Engineering* 7, 3 (1995), 436–453.
[7] Sherif Sakr, Angela Bonifati, Hannes Voigt, Alexandru Iosup, Khaled Ammar, Renzo Angles, Walid Aref, Marcelo Arenas, Maciej Besta, Peter A Boncz, et al. 2021. The future is big graphs: a community view on graph processing systems. *Commun. ACM* 64, 9 (2021), 62–71.
[8] Muhammad Usman, Mian Ahmad Jan, Xiangjian He, and Jinjun Chen. 2019. A survey on big multimedia data processing and management in smart cities. *ACM Computing Surveys (CSUR)* 52, 3 (2019), 1–29.
[9] Chuangxian Wei, Bin Wu, Sheng Wang, Renjie Lou, Chaoqun Zhan, Feifei Li, and Yuanzhe Cai. 2020. AnalyticDB-V: a hybrid analytical engine towards query fusion for structured and unstructured data. *Proceedings of the VLDB Endowment* (Aug 2020), 3152–3165.
[10] Zihao Zhao, Zhihong Shen, Along Mao, Huajin Wang, and Chuan Hu. 2023. PandaDB: An AI-Native Graph Database for Unified Managing Structured and Unstructured Data. In *International Conference on Database Systems for Advanced Applications*. Springer, 669–673.