A New Graph-based Approach for Building Information Querying

Junxiang Zhu, jz652@cam.ac.uk
Department of Engineering, University of Cambridge, Cambridge, United Kingdom

Nicholas Nisbet, N.Nisbet@ucl.ac.uk
Bartlett School of Sustainable Construction, University College London, London, United Kingdom

Mengtian Yin, my424@cam.ac.uk
Department of Engineering, University of Cambridge, Cambridge, United Kingdom

Jinying Xu, jx314@cam.ac.uk
Department of Engineering, University of Cambridge, Cambridge, United Kingdom

Ran Wei, r.wei5@lancaster.ac.uk
School of Computing and Communications, Lancaster University, Lancaster, United Kingdom

Ioannis Brilakis, ib340@cam.ac.uk
Department of Engineering, University of Cambridge, Cambridge, United Kingdom

Abstract
Graph is potential for building information storage and querying. A new graphical representation of IFC (Industry Foundation Classes) data has been developed, referred to as IFC-Graph. However, many aspects about querying information from this graphically structured data are not clear. This study aims to explore the querying of information from IFC-Graph by using graph query language. One IFC model was converted into IFC-Graph and used for validation. Experiments were carried out to query individual instances and their relations by using Cypher. The results show that a) IFC-Graph can support building information querying, b) Cypher can be used to query individual instances and the complex relations between instances, such as spatial structure and space boundary. C) Cypher is more efficient in query relations compared with IFC parser. This study can contribute to the development and usage of digital twin by providing an effective way for building information querying.

Keywords: Digital Model; Industry Foundation Classes (IFC); Graph; Knowledge Discovery, Geographic Information System (GIS); Building Information Modelling (BIM)

1 Introduction

Digitalisation is a trend over the world across disciplines. While digitalisation brings benefits to effective decision making, one of the challenges coming along is the efficient query of digital data, especially for areas with multiple stakeholders who use heterogeneous data, such as the construction industry (Nepal et al., 2012, Chi et al., 2017, Tan et al., 2023). With the digitalisation of construction projects, practitioners are faced with the difficulty in efficiently finding the information they need. For example, it is estimated that in Australia, construction workers spend about 4.9 hours hunting down project data each week, resulting in a loss of $15.6 billion in labour costs (PlanGrid, 2018).

The challenge in efficient information querying in the construction industry comes from the use of heterogeneous data that are in various formats and forms (e.g., individual files and isolated databases). This leads to the well-known interoperability issue in the Architecture, Engineering, Construction, and Operation (AECO) domain (Zhu et al., 2022, Zhu et al., 2020). In recent years,

with the advance of data science in developing new data formats and new database systems, it is now possible to manage the interoperability issues. However, it is still a challenge to effectively query information from these digital building models due to their well-known interconnected structure that is full of relationships and the growing size of the Industry Foundation Classes (IFC, ISO16739-1) specification. This is an outstanding problem in the bigger context of digital twins (Sacks et al., 2020, Lu and Brilakis, 2019) and smart cities (Zhu and Wu, 2021, Zhu et al., 2021, Silva et al., 2018).

One of the promising techniques to address this issue is graph technology, which is good at handling relationships (Robinson et al., 2015, Angles and Gutierrez, 2008). By converting digital building models into graphs, information in the digital models can be effectively retrieved using advanced graph traversal algorithms, even for the hidden relationship information (Zhu et al., 2023). Zhu et al. developed a method to fully convert building information into labelled property graphs (LPG), referred to as IFC-Graph (Zhu et al., 2023). However, this new graphical representation of building information has not been sufficiently assessed. Therefore, the aim of this study is to explore the querying of information from IFC-Graph.

The remainder of this paper is organised as follows. Section 2 introduces the existing methods for building information querying. Section 3 presents the querying of IFC-Graph by using graph query language. Section 4 is about the validation of this study. Discussions are presented in Section 5, and Section 6 conclusions the paper.

2 Literature review

2.1 Building information querying

Information querying refers to the process of finding the required information from datasets, and a query is a statement requesting the retrieval of information (Silberschatz et al., 2019). In general, there are two approaches for querying building information, depending on the environments used, i.e., file-processing environment or database environment.

2.1.1 The file-based environment

File-based environment queries building information from individual files that contains digital models, such as IFC-SPF (STEP Physical File). There are two subtypes for this environment. The first is using existing file parsers, such as IfcOpenShell (IfcOpenShell, 2024) and IfcEngine DLL (RDF ltd., 2024) for IFC-SPF and the general XML (Extensible Markup Language) parser XQuery (W3Schools, 2024) for IFC-XML (Nepal et al., 2012). Parsers provide capability in effective building information access, but not efficient building information querying. As stated by (Silberschatz et al., 2019), conventional file-processing environments do not allow a convenient and efficient way for needed data to be retrieved. However, this is the most commonly used way for building information querying, because IFC is currently optimised for file-based exchange (van Berlo et al., 2021). The second is using customised query languages, such as the Query Language for Building Information Models (QL4BIM) by (Daum and Borrmann, 2014) and the Visual Code Checking Language (VCCL) by (Preidel et al., 2017). These query languages are domain specific. For example, QL4BIM is for querying spatial topology of building elements after processing and analysing their geometry (i.e., boundary representation), while VCCL provides a visual way to conduct code compliance check for produced building models.

2.1.2 The database environment

Compared to file-based environments, database environments provide better support for information querying by offering well-designed query languages, such as SQL (Structured Query Language) for relational databases and graph query languages (such as Cypher) for graph databases (Date, 1989, Neo4j Inc., 2024b). This has also inspired buildingSMART, i.e., the organisation managing the IFC standard, to develop a query language and an object-based API (application programming interface) for IFC when planning for IFC 5 (van Berlo et al., 2021).

Among these two general types of database management systems, previous studies mainly used NoSQL (Not Only SQL) database management system, especially graph database, to handle building information. This is because the conventional SQL Database Management System (DBMS) cannot effectively handle relationships (Robinson et al., 2015), thus not efficient

information querying as well. In contrast, graph database is tailored for handling relationships, such as the friend-of-a-friend relationship in social network (Miller, 2013). Such a capability of graph was also used to reveal the relationship between building elements, which is an important part of the semantic information.

When talking about graph, RDF (Resource Description Framework) is an inevitable topic, which is for constructing the Semantic Web and considered to be graph based (Pauwels and Terkaj, 2016, Pauwels et al., 2017, Lei et al., 2022). The databases for storing RDF triples are referred to as RDF Triplestore (ontotext, 2024). This study confines graph database to **native** graph database that uses native graph storage (using property graph as data storage structure) and native graph processing (using index-free adjacency), as defined by (Robinson et al., 2015). The RDF Triplestores are not within this scope.

2.2 Challenges in graph-based building information querying

While graph database provides a good capability in handling relationships and querying graphs, the problem was that there was no full graphical representation of building data available for further assessment. This was mainly due to the lack of a method to fully and automatically convert IFC data into graph (Zhu et al., 2023). There were studies trying to convert IFC data into graphs. For example, (Ismail et al., 2017, Ismail et al., 2018) examined the use of graph database in storing building information and the use of graph-based query for pathfinding. However, methods developed by previous studies for data conversion and graph generation was partial, just right to meet their own information needs for specific applications, such as data analysis and visualisation (Nahar, 2017, Lather et al., 2020, Donato, 2017, Ge et al., 2019). The workflow for converting and using building information via graph by previous studies is presented in Figure 1.

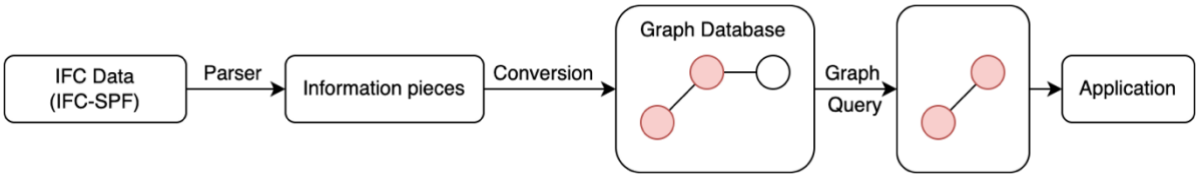

Figure 1 Graph generation and graph querying by previous studies.

The above problem has been addressed by (Zhu et al., 2023). They developed an algorithm to convert IFC-SPF files fully and automatically into labelled property graphs (LPGs), which are referred to as IFC-Graph. IFC-Graph is a new form of IFC information, and many aspects have not been well investigated, such as a) what is the difference in querying general graphical data and IFC-Graph? b) what information can be queried from IFC-Graph and what information can be quired more efficiently than other information querying means. These questions are critical for effectively retrieving information from IFC-Graph, or more broadly, any graphical building information.

3 The new graph-based building information querying

Based on the above identified problems, this study explored the querying of IFC-Graph by using Cypher, which is a generic graph query language (GQL) developed by Neo4j. Cypher is the key inspiration for the international standard for graph query language, i.e., ISO/IEC 39075:2024 (Neo4j Inc., 2024d, ISO, 2024). The joint use of graphical building information and graph query language can create a new paradigm of querying building information, as presented in Figure 2.

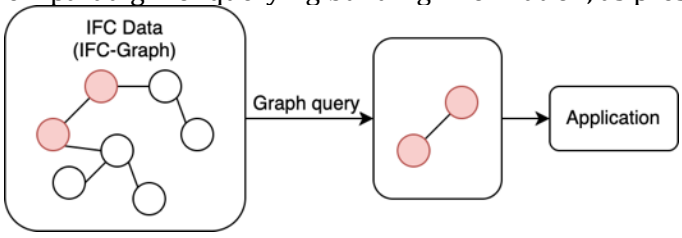

Figure 2 Graph-based IFC information querying.

3.1 IFC data model and IFC data

A good understanding of the underlying data model is essential for effective information querying. The relationship between IFC data model and IFC data is that the former is a higher level of abstraction of IFC data, which provides useful reference when querying IFC data. In IFC, the most basic elements are entities. Entities defines objects (IfcObjectDefinition), properties (IfcPropertyDefinition), and relationships (IfcRelationship). Object entities define types of elements, their attributes, while relationship entities define the connections between object entities and/or property entities. A detailed description of the IFC data model can be found in (buildingSMART International, 2017). The relationship entities in IFC are like bridges connecting entities, implicitly making the IFC data model a graph. Due to the existence of various relationship entities in IFC, many useful types of information are available in IFC data, such as property sets, space boundary, and the spatial structure.

3.2 LPG-based IFC data

IFC-Graph is a full graphical representation of IFC data based on LPG. In the study by (Zhu et al., 2023), the IFC elements (i.e., entities and attributes) are mapped to graph elements (i.e., nodes and edges) using the mapping presented in Table 1. To distinguish the attributes in IFC, the concepts of intrinsic attribute and extrinsic attribute are borrowed from model-driven engineering (MDE) (Brambilla et al., 2017). Intrinsic attributes are attributes having only primitive data values, while extrinsic attributes are those representing relationships. Entities are mapped to nodes, intrinsic attributes to properties of nodes, and extrinsic attributes to edges. An advantage of using such a mapping is the preservation of the original IFC data structure.

Table 1 Mapping IFC elements (entities, attributes) to graph elements (nodes, edges, properties).

	From (IFC)	To (IFC-Graph)
Entity	Objects	Nodes
	Relationships	Nodes
	Property sets	Nodes
Attributes	Intrinsic attributes	Properties of nodes
	Extrinsic attributes	Edges

3.3 Graph-based building information querying

3.3.1 The graph database and query language

In this study IFC-Graphs were stored in neo4j, a native graph database management system that uses property graph for data storage and data querying (Robinson et al., 2015). Cypher is the native query language for neo4j, and a key concept in graph query is the use of patterns, just like the 'pattern' concept in regular expression for extracting textual information (Thompson, 1968). Patterns can be used to represent human's knowledge about an area, hence, graph query language is also referred to as inferencing query language (Robinson et al., 2015).

3.3.2 Query patterns in Cypher

Table 2 presents the query patterns in Cypher for querying nodes, relationships, and paths. Nodes are indicated by '( )', which can include labels following a ': ', and properties are within a pair of curly brackets. Edges are denoted by '— —', which can include labels as well, and arrows can be used to indicate the direction of the connection. Paths are formed by a series of nodes and edges. A more detailed instruction on using Cypher can be found in (Neo4j Inc., 2024a).

Table 2 The default basic patterns for querying nodes, relationships, and paths in graph.

	Type	The basic pattern	Query
1	Nodes	`(n:Label{property:value})`	`match (n:Label{property:value}) return n`
2	Edges	`<-[r:Label]->`	`match (n1)<-[r:Label]->(n2) return r`
3	Paths	`(n1)<-[r]->(n2)`	`match p=(n1)<-[r]->(n2) return p`

3.3.3 Building knowledge expressed by graph patterns

With the graph patterns, human knowledge about buildings can be expressed and used to query IFC data. For example, it is a common sense that a room (or space in IFC) is surrounded by floors, walls, windows, and doors. Such knowledge can be expressed by the pattern presented in Listing 1. By doing this, human knowledge about buildings is represented by patterns and can be directly used in building information query. The patterns for querying the spatial structure, spatial containment, and space boundary are presented in Table 3.

Listing 1 Pattern for representing knowledge of space boundary.

```
1   (:IfcSpace)--(:IfcRelSpaceBoundary)--(:IfcSlab) OR
2   (:IfcSpace)--(:IfcRelSpaceBoundary)--(:IfcWall) OR
3   (:IfcSpace)--(:IfcRelSpaceBoundary)--(:IfcWindow) OR
4   (:IfcSpace)--(:IfcRelSpaceBoundary)--(:IfcDoor)
```

Table 3 Patterns for querying spatial structure, spatial containment, and space boundary.

	Building information	Pattern
1	Spatial Structure	`(:IfcSpatialStructureElement)--(:IfcRelAggregates)--(:IfcSpatialStructureElement)`
2	Spatial Containment	`(:IfcBuildingElement)--(:IfcRelContainedInSpatialStructure)--(:IfcSpatialStructureElement) return p`
3	Space Boundary	`(:IfcSpace)--(:IfcRelSpaceBoundary)--(:IfcBuildingElement)`

4 Validation

4.1 Data

The Institute model from Karlsruhe Institute of Technology (KIT) (Figure 3 (a)) was used for validation. This model was converted into IFC-Graphs (Figure 3 (b)) using the method developed by (Zhu et al., 2023). An overview of this model and IFC-Graph is presented in Table 4. Scripts have been developed to carried out the queries of individual instances and relations.

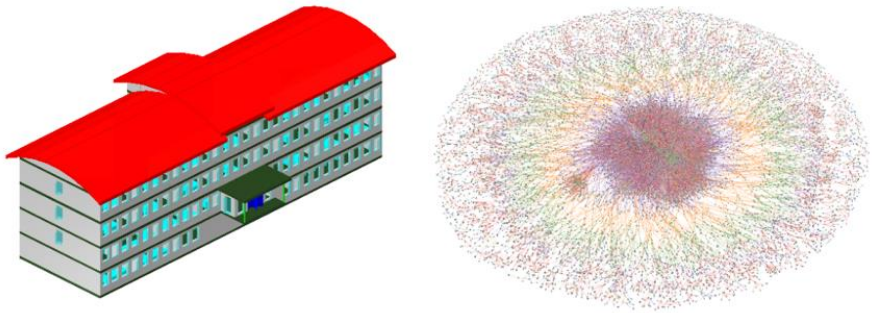

Figure 3 IFC models and IFC-Graphs used for validation.

Table 4 Overview of the IFC-SPF and IFC-Graph for the Institute model.

	IFC-SPF			IFC-Graph	
Model	File Size	Entities/Instances	Version	Nodes	Edges
Institute	10.9 MB	189/147712	IFC4	194845	285504

4.2 Retrieving instances of entities

Instances of entities were queried by entity type using both IfcOpenShell (within Python) and Cypher. The codes for querying are listed in Table 5. The queried entities include IfcObjectDefinition, IfcRelationship, and IfcPropertyDefinition, i.e., the immediate child entities of IfcRoot.

Table 5 Queries used for validation.

	Tool	Query
1	IfcOpenShell	`def get_instance(IfcClass): n=ifcopenshell.by_type(IfcClass) return len(n)`
2	Cypher	`match (n:IfcClass) return count(n)`

The result is presented in Table 6, which shows that Cypher can retrieve the same individual instances of entities, just like IfcOpenShell. The query time (in milliseconds) of IfcOpenShell is dependent on the number of instances to be returned, while the query time of Cypher is relatively stable. It should be noted that it usually takes longer for the first run of query because of the time needed to interpret the query command by the graph DBMS. The query time presented in Table 6 for Cypher is the average query time of 10 iterations of the query.

Table 6 Query results of individual instances of entities.

	Entity	Number of Instances		Query Time (Milliseconds)	
		IfcOpenShell	Cypher	IfcOpenShell	Cypher
1	Object	1210	1210	5.281	1.971
2	Relationship	6374	6374	22.185	2.381
3	Property	4446	4446	8.298	2.942

4.3 Querying relations of entities

Relations between entities are substantial part of IFC (van Berlo et al., 2021). The common relations in IFC include spatial structure (i.e., the hierarchy of spatial entities), spatial containment (i.e., the assignment of physical entities into the spatial hierarchy), and space boundary (i.e., the relationship between spaces and physical entities that bound them). The patterns listed in Table 3 were used to design queries for these relations, an example query is given in Listing 2 using spatial structure. The query results are presented in Table 7. Each query was run 10 times to get the maximum, mean, and minimum query time.

Listing 2 Python codes for querying spatial structure.

```
1     def get_spatial_structure(graph):
2         cypher = 'MATCH p=(:IfcSpatialStructureElement)--(:IfcRelAggregates)--
3     (:IfcSpatialStructureElement) RETURN p'
4         result = graph.run(cypher)
5         return result
```

Table 7 Query results of relations.

	Information	Graph		Query Time (Seconds)		
		Nodes	Edges	Max	Mean	Min
1	Spatial Structure	97	192	2.813	1.857	1.601
2	Spatial Containment	458	906	0.613	0.478	0.381
3	Space Boundary	1504	7936	1.159	1.086	1.013

It is reasonable to assume that much more work would be required when IfcOpenShell is used to retrieve the same information, due to the manual traversal of the query path. The queried relation graphs from the Institute model for spatial structure, spatial containment, and space boundary are presented in Figure 4.

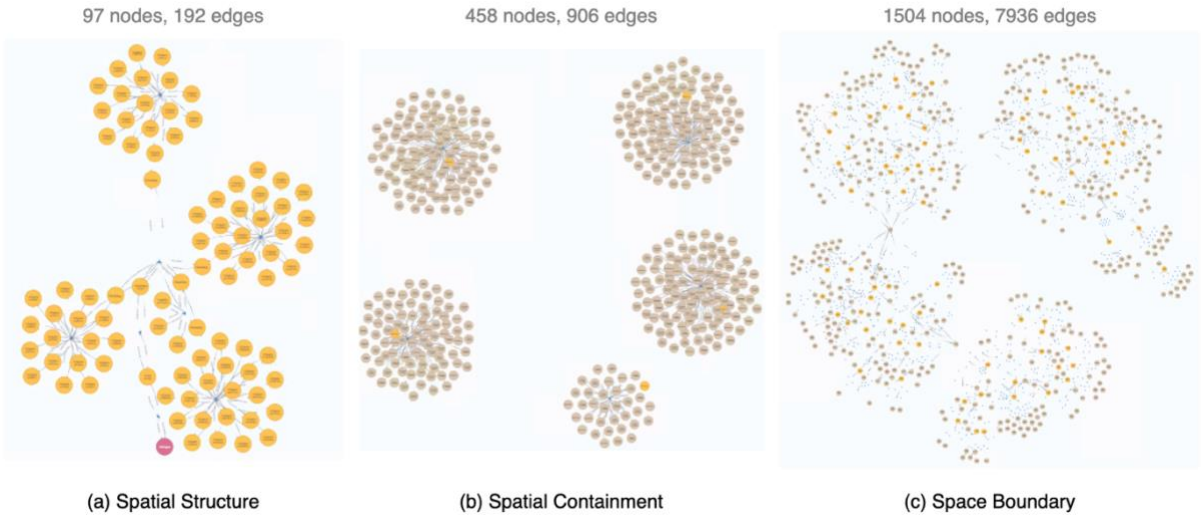

Figure 4 The queried relation graphs from the Institute model for (a) spatial structure, (b) spatial containment, and (c) space boundary.

5 Discussion

This study explored the use of Cypher to query building information from IFC-Graph. IFC-Graph, together with Cypher, can reveal and visualise hidden information in building models by using graph pattern matching, which can be used for graph-based knowledge discovery (Neo4j Inc., 2024c).

5.1 The expressiveness of graph pattern

To further demonstrate the expressiveness of graph pattern, a simple comparison was carried out. Table 8 shows the codes needed to retrieve the spatial structure by using Cypher, IfcOpenshell, and SPARQL from IFC-Graph, IFC-SPF, RDF in graph form and RDF in Turtle, respectively. Query codes using graph pattern matching are more compact, thus more expressive. This is even the case for RDF. With Cypher and IFC-Graph, useful relation information can be easily and effectively retrieved, rather than writing long queries using IFC parsers. An additional example of querying IFC-XML using XPath is also provided in Table 8 for reference.

Table 8 Codes needed to retrieve the spatial structure.

	Format	Language	Query
1	IFC-Graph	Cypher	# Get all sites, buildings, storeys, spaces, and their relation `match p=( )--(:IfcRelAggregates)--( ) return p`

2	IFC-SPF	IfcOpenShell	# Only get the first site, the first building, the first building storey, the first space, and their relation; repeat the process to get all the instances of IfcSpatialStructureElement and their relations.
			`project = ifc_file.by_type('IfcProject')[0]`
			`site_1 = project.IsDecomposedBy[0].RelatedObjects[0]`
			`building_1 = site_1.IsDecomposedBy[0].RelatedObjects[0]`
			`storey_1 =`
			`building_1.IsDecomposedBy[0].RelatedObjects[0]`
			`space_1 = storey_1.IsDecomposedBy[0].RelatedObjects[0]`

	Format	Language	Query

3	RDF (Graph)	Cypher	`match p=()--(r)--()`
			`where r.uri contains 'IfcRelAggregates'`
			`return p`

4	RDF (Turtle)	SPARQL	`PREFIX ifcowl:<http://ifcowl.openbimstandards.org/IFC4#>`
			`Select ?Project ?Site ?Building ?BuildingStorey ?Space`
			`WHERE {`
			`?Project a ifcowl:IfcProject`
			`?Relaggregate1 relatingObject_IfcRelAggregates ?Project`
			`?Relaggregate1 relatedObjects_IfcRelAggregates ?Site`
			`?Relaggregate2 relatingObject_IfcRelAggregates ?Site`
			`?Relaggregate2 relatedObjects_IfcRelAggregates ?Building`
			`?Relaggregate3 relatingObject_IfcRelAggregates ?Building`
			`?Relaggregate3 relatedObjects_IfcRelAggregates ?BuildingStorey`
			`?Relaggregate4 relatingObject_IfcRelAggregates ?BuildingStorey`
			`?Relaggregate4 relatedObjects_IfcRelAggregates ?Space}`

5	IFC-XML	XPath	`//IfcRelAggregates/RelatedObjects/*[name()='IfcSite' or name()='IfcBuilding' or name()='IfcBuildingStorey' or name()='IfcSpace']/concat(../../RelatingObject/*/@ref,' , ', @ref)`

5.2 Limitation and future work

Even though this study has demonstrated some features of IFC-Graph and Cypher, this is just a preliminary exploration on the use of graph for generic building information querying. More work should be carried out in the future. One of the potential works is the querying of other relations in IFC, for example, the space connectivity (that can be used in emergency response) and the element connectivity (that can be used to support structural loading and construction sequencing analyses).

Limitations of the IFC-Graph were noticed as well regarding the usability of the queried information and the simplification of graph. First, the queried graphical building information is raw and contains certain unnecessary information, such as the link between an object node and a relationship node. Such information needs further processing before it can be practically used. Second, graphs can be further simplified to make them more usable in practice. This involves the establishment of new data models that are tailored for specific use cases. The new data models are usually simpler than the IFC specification and result in smaller graphs.

6 Conclusion

Graph has been recognised as a highly potential form for building information storage and querying. This study explored the use of graph query language for querying building information from IFC-Graph, which is a new graph form of building information based on LPG. A building model in IFC-SPF was converted into IFC-Graph, and Cypher was used to query individual instances and relations from IFC-Graph. The main findings and outcomes of this study are as follows. The graphical representation of building information (IFC-Graph) can make full use of graph algorithms to retrieve building information from interconnected datasets, and graph database has better performance for information querying than file-processing parsers, especially for retrieving relations. This study is mainly for sophisticated users who have a strong

need to retrieve and use building information from IFC in a more effective and efficient manner. Further work is required to make the queried graph more usable in practice, for example, by simplifying graph through eliminating attributes and some objectified relationships.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 101034337. We would like to thank Dr. Lavindra De Silva, Dr. Pieter Pauwels, Dr. Stefano Cavazzi, Yogesh Patel, Matt Peck, Katrin Johannesdottir, George Economides, Ajay Gupta, Peter Lindgren, and Alex Gillies for their contributions to this paper.

References

Angles, R. & Gutierrez, C. (2008). Survey of graph database models. *ACM Computing Surveys (CSUR),* 40**,** 1-39.

Brambilla, M., Cabot, J. & Wimmer, M. (2017). *Model-driven software engineering in practice*, Morgan & Claypool Publishers.

Buildingsmart International. (2017). *Industry Foundation Classes 4.0.2.1* [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/ [Accessed 10 April 2024].

Chi, H.-L., Chai, J., Wu, C., Zhu, J., Wang, X. & Liu, C. (2017). Scaffolding progress monitoring of LNG plant maintenance project using BIM and image processing technologies. 2017 International Conference on Research and Innovation in Information Systems (ICRIIS), 2017. IEEE, 1-6.

Date, C. J. (1989). *A Guide to the SQL Standard*, Addison-Wesley Longman Publishing Co., Inc.

Daum, S. & Borrmann, A. (2014). Processing of topological BIM queries using boundary representation based methods. *Advanced Engineering Informatics,* 28**,** 272-286.

Donato, V. (2017). Towards design process validation integrating graph theory into BIM. *Architectural Engineering and Design Management,* 13**,** 22-38.

Ge, G., Yue-Mei, Z., Han, L., Zhi, L. & Ming, G. (2019). Research on IFC Model Checking Method Based on Knowledge Base. *Journal of Graphics,* 40**,** 1099.

Ifcopenshell. (2024). *IfcOpenShell* [Online]. Available: http://ifcopenshell.org/ [Accessed 16 April 2024].

Ismail, A., Nahar, A. & Scherer, R. (2017). *Application of graph databases and graph theory concepts for advanced analysing of BIM models based on IFC standard* [Online]. Nottingham, UK. Available: https://www.researchgate.net/publication/318600860_Application_of_graph_databases_and_graph_theory_concepts_for_advanced_analysing_of_BIM_models_based_on_IFC_standard [Accessed 16 April 2024].

Ismail, A., Strug, B. & Ślusarczyk, G. (2018). Building knowledge extraction from BIM/IFC data for analysis in graph databases. International Conference on Artificial Intelligence and Soft Computing, 3-7 June, 2018 2018 Zakopane, Poland. Springer, 652-664.

Iso. (2024). *ISO/IEC 39075:2024 Information technology — Database languages — GQL* [Online]. Available: https://www.iso.org/standard/76120.html [Accessed 15 May 2024 2024].

Lather, J. I., Logan, T., Renner, K. & Messner, J. I. (2020). Implementation and evaluation of generative layout options using the graph theoretical approach for a hospital layout problem. *Journal of Computing in Civil Engineering,* 34**,** 04020014.

Lei, X., Wu, P., Zhu, J. & Wang, J. (2022). Ontology-Based Information Integration: A State-of-the-Art Review in Road Asset Management. *Archives of Computational Methods in Engineering,* 29**,** 2601-2619.

Lu, R. & Brilakis, I. (2019). Digital twinning of existing reinforced concrete bridges from labelled point clusters. *Automation in construction,* 105**,** 102837.

Miller, J. J. (2013). Graph database applications and concepts with Neo4j. Proceedings of the southern association for information systems conference, Atlanta, GA, USA, 2013. 141-147.

Nahar, A. (2017). *Applying graph theory concepts for analyzing BIM models based on IFC standards.* Master, Dresden University of Technology.

Neo4j Inc. (2024a). *Cypher Manual* [Online]. Available: https://neo4j.com/docs/cypher-manual/current/introduction/ [Accessed 10 April 2024].

Neo4j Inc. (2024b). *Introduction - Cypher Manual* [Online]. Available: https://neo4j.com/docs/cypher-manual/current/introduction/ [Accessed 16 April 2024 2024].

Neo4j Inc. (2024c). *The Power of Graph Data Science* [Online]. Available: https://neo4j.com/whitepapers/the-power-of-graph-data-science/ [Accessed 17 April 2024].
Neo4j Inc. (2024d). *What is openCypher?* [Online]. Available: https://opencypher.org/ [Accessed 16 April 2024].
Nepal, M. P., Staub-French, S., Pottinger, R. & Webster, A. (2012). Querying a building information model for construction-specific spatial information. *Advanced Engineering Informatics,* 26**,** 904-923.
Ontotext. (2024). *What is an RDF Triplestore* [Online]. Available: https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/ [Accessed 12 Mar 2024 2024].
Pauwels, P., Krijnen, T., Terkaj, W. & Beetz, J. (2017). Enhancing the ifcOWL ontology with an alternative representation for geometric data. *Automation in Construction,* 80**,** 77-94.
Pauwels, P. & Terkaj, W. (2016). EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction,* 63**,** 100-133.
Plangrid. (2018). *Construction Firms in Australia and New Zealand Lose $36.5 Billion in Labour Costs on Unproductive Work* [Online]. Available: https://www.autodesk.com/blogs/construction/construction-firms-australia/#:~:text=Each%20construction%20worker%20on%20a,management%20consulting%20firm%20FMI%20Corporation. [Accessed 9 April 2024].
Preidel, C., Daum, S. & Borrmann, A. (2017). Data retrieval from building information models based on visual programming. *Visualization in Engineering,* 5**,** 1-14.
Rdf Ltd. (2024). *IFC Engine* [Online]. Available: https://rdf.bg/product-list/ifc-engine/ [Accessed 16 April 2024].
Robinson, I., Webber, J. & Eifrem, E. (2015). *Graph databases: new opportunities for connected data,* Sebastopol, CA, the United States, O'Reilly Media, Inc.
Sacks, R., Brilakis, I., Pikas, E., Xie, H. S. & Girolami, M. (2020). Construction with digital twin information systems. *Data-Centric Engineering,* 1**,** e14.
Silberschatz, A., Korth, H. F. & Sudarshan, S. (2019). *Database System Concepts,* New York, McGraw-Hill Education.
Silva, B. N., Khan, M. & Han, K. (2018). Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities. *Sustainable cities and society,* 38**,** 697-713.
Tan, Y., Liang, Y. & Zhu, J. (2023). CityGML in the Integration of BIM and the GIS: Challenges and Opportunities. *Buildings,* 13**,** 1758.
Thompson, K. (1968). Programming techniques: Regular expression search algorithm. *Communications of the ACM,* 11**,** 419-422.
Van Berlo, L., Krijnen, T., Tauscher, H., Liebich, T., Van Kranenburg, A. & Paasiala, P. (2021). Future of the industry foundation classes: towards IFC 5. 38th International Conference of CIB W78, 2021. 123-137.
W3schools. (2024). *XQuery Tutorial* [Online]. Available: https://www.w3schools.com/xml/xquery_intro.asp [Accessed 18 April 2024].
Zhu, J., Chong, H.-Y., Zhao, H., Wu, J., Tan, Y. & Xu, H. (2022). The Application of Graph in BIM/GIS Integration. *Buildings,* 12**,** 2162.
Zhu, J., Tan, Y., Wang, X. & Wu, P. (2020). BIM/GIS integration for web GIS-based bridge management. *Annals of GIS***,** 1-11.
Zhu, J. & Wu, P. (2021). Towards Effective BIM/GIS Data Integration for Smart City by Integrating Computer Graphics Technique. *Remote Sensing,* 13**,** 1889.
Zhu, J., Wu, P. & Anumba, C. (2021). A Semantics-Based Approach for Simplifying IFC Building Models to Facilitate the Use of BIM Models in GIS. *Remote Sensing,* 13**,** 4727.
Zhu, J., Wu, P. & Lei, X. (2023). IFC-graph for facilitating building information access and query. *Automation in Construction,* 148**,** 104778.