Watanuki T. and Yabuki N. (2017). "Data Management Methods for a 3D Product Model in Graph Database." In: *LC3 2017: Volume I – Proceedings of the Joint Conference on Computing in Construction (JC3)*, July 4-7, 2017, Heraklion, Greece, pp. 751-758. https://doi.org/10.24928/JC3-2017/0025.

DATA MANAGEMENT METHODS FOR A 3D PRODUCT MODEL IN GRAPH DATABASE

Tsutomu Watanuki[1] and Nobuyoshi Yabuki[2]

Abstract: In Building Information Modeling (BIM), a product model of a building includes huge data which is proceeded in its whole lifecycle. Many stakeholders need to share the product model on the Internet in order to work cooperatively. However, it is difficult to share the product model on the Internet because the amount of product model data is huge and has a very deep hierarchical structure. Thus, in this research, we propose data management methods to store and obtain data of product models. In these methods, Industry Foundation Classes (IFC) data is converted and stored in a graph DBMS classified as NoSQL, which stands for Not only SQL.

Finally, good results have been obtained in queries from the graph database in which product models are stored. And we were able to extract representation data of a storey from a product model of a building which has 5 stories.

Keywords: BIM, Graph database, NoSQL, IFC, Graph theory

1 INTRODUCTION

In BIM, it is necessary to share 3D product models that include huge volumes of data between all relevant parties in large scale projects. Today, files are passed back and forth to give and receive product models via a network, using email or a file-transfer service or the like. However, there is a problem in that file transfer can take a long time because of the enormous volume of data being transferred. Also, there is a problem in that it is difficult for work to move forward concurrently by multiple workers because files tend to be scattered between the many people involved in a project.

To solve these problems, information-sharing systems for BIM were developed by some researchers and developers. Das et al developed Social BIMCloud (Das 2015) using Apache Casandra, which is column oriented DBMS, also Breetz et al did BIM Server (Breetz 2010) using Oracle BerkleyDB, which is key-value store type DBMS. In this way, several kinds of DBMS were used to store 3D product models which are described by the IFC schema, however all kinds of DBMS were not attempted to do it. Therefore, it does not become clear which DBMS is suitable to store and obtain data of IFC product models.

A graph database comes to be used for database system development recently. The database is designed for data whose relations are well represented as a graph consisting of elements interconnected with a number of relations between them. The authors thought a graph database is suitable to store data of IFC product models, because a property graph model, which is a data model of the graph database, and IFC product model has high similarity.

[1] Technical Innovation Center, Kawada Technosystem Co., Ltd, Tokyo, Japan, t-watanuki@kts.co.jp
[2] Professor, Division of Sustainable Energy and Environmental Engineering, Graduate School of Engineering, Osaka University, Japan, yabuki@see.eng.osaka-u.ac.jp

Thus, in this research, a product model management method stored in a graph DBMS was developed, supposing the building of an information-sharing system that uses a graph DBMS.

2 Graph Database VS Relational Database

2.1 Target Product Model

The IFC schema was designed based on object-oriented modeling; the concept of all objects including walls and doors and the like that constitute a building are defined as classes. Combinations of instances that embody these classes is the IFC product model, and they have a deep hierarchical structure. Figure 1 shows a portion of the model that described the IFC product model of a five-storey building using UML.

2.2 Conversion of IFC Product Model

2.2.1 Conversion to Relational Model

It is needed to convert the IFC product model into a relational model, which is the relational DBMS data model, in order to store the IFC product model in the relational DBMS. With the relational model, data is managed in a structure that is similar to a table called a relation. It is possible to include multiple attributes in one relation. Aggregates of values of each attribute are a tuple; it is possible to implement relational algebra operations on multiple relations.

With this relational model, it is presumed that data is normalized to reduce redundancy; data is separated into multiple relations and stored. Therefore, define the relation and the attributes for each class of the IFC schema, and store in corresponding relations by separating into each instance.

2.2.2 Conversion to Property Graph Model

It is needed to convert the IFC product model into a property graph model (Robinson et al 2013) which is a data model of graph DBMS, in order to store the IFC product model in the graph DBMS. It is possible to convert an IFC product model into a property graph model while maintaining the structure substantially as it is by associating one instance of an IFC product model to one node, and associating relationships between instances to edges. Also, attribute values of each instance are retained as properties of the node.

2.3 Performance

Figure 2 shows the time required for obtaining data with differing hierarchies, with the IFC product model stored in two DBMS's, namely an SQL Server 2008 R2 (Microsoft n.d.), which is relational DBMS, and Neo4j (Neo Techonology n.d.), which is graph DBMS. Processing time greatly increases for the SQL Server as the hierarchy increases, but with Neo4j, it is possible to get data with a substantially fixed amount of time regardless of the number of depth of the data to be obtained.

2.4 Selection of Database

As shown in Figure 1, because the hierarchical structure of the IFC product model is deep, the relational DBMS in which performance deteriorates for data deep in the hierarchy was not the subject of this research. This research focused on graph DBMS's that do not experience such degraded performance and on a data management method in a graph DBMS.

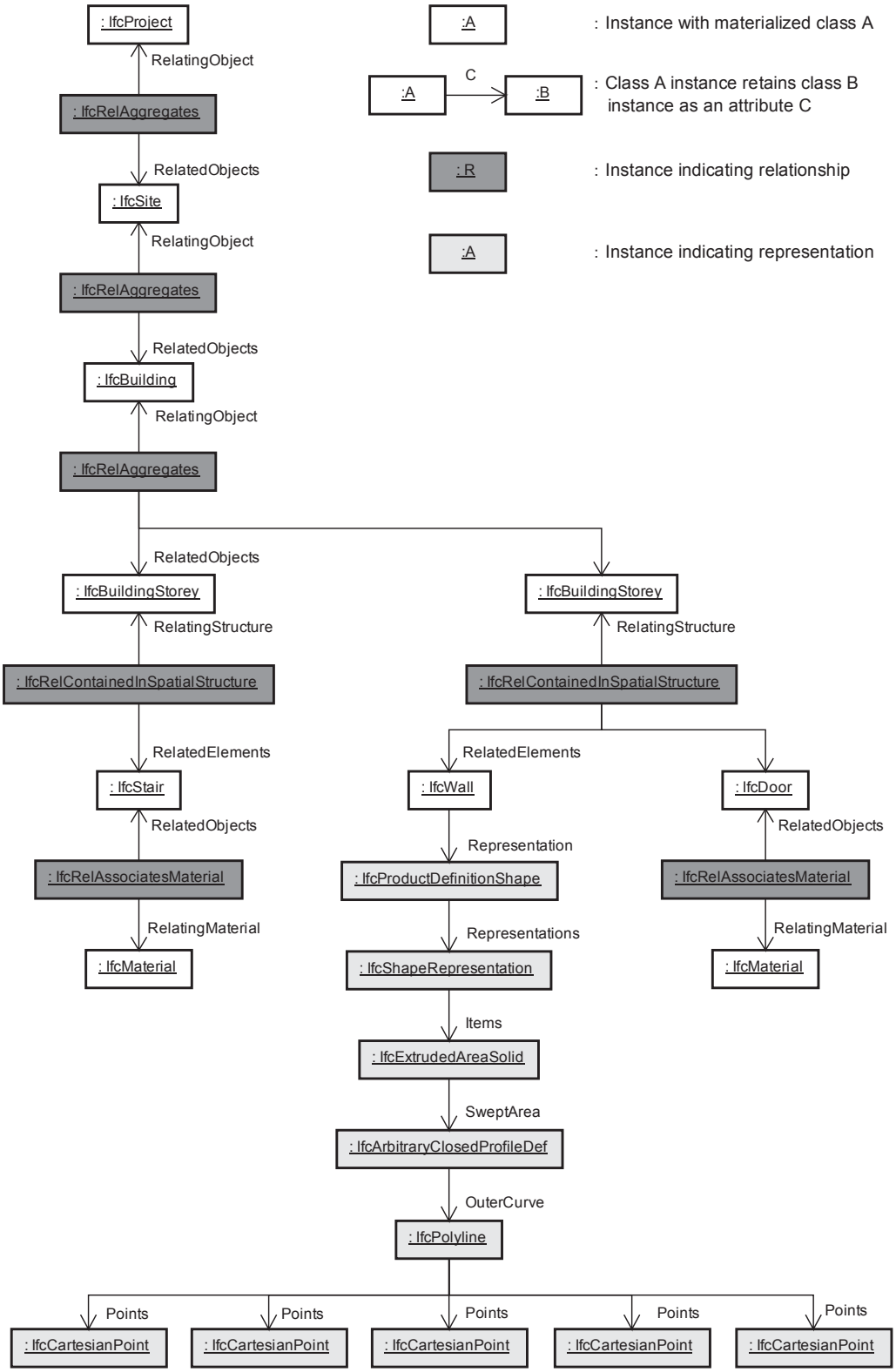

Figure 1: Portion of IFC Product Model (UML Object Diagram)

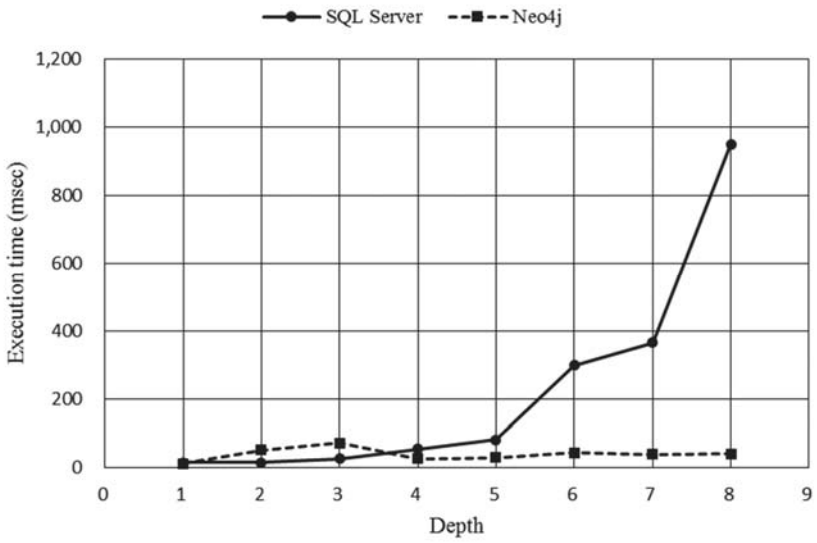

Figure 2: Data Obtaining Time by the Number of Layers in a Hierarchy

3 Data Management Methods in Graph Database

3.1 Partial Extraction Method

The authors studied a method for extracting only a portion of a model from an IFC product model stored in the graph DBMS. This research focused on studying a data extraction method for shapes at a specific location, which has high demand for use in product models in BIM.

3.2 Partial Extraction of Shape Data

With the IFC schema, an instance having a physical shape is defined to inherit the IfcProduct class. Therefore, it is acceptable to extract an instance that is associated to an instance in a class derived from the IfcProduct (the thinly gray node in Figure 1). This becomes the partial graph aggregate using the IfcProduct-derived class instance as the starting node, and the IfcCartesianPoint instance as the terminal node.

3.2.1 Partial Extraction of Specific Location Data

Next, consideration is given to extracting data for a specific location, for example data for only a specific floor. As can be understood by looking at Figure 1, spatial elements such as a floor are expressed using a hierarchic structure in the IFC schema. Therefore, processes are implemented to extract all elements included in the targeted spatial elements. The spatial elements and physical elements included therein have an IfcRelContainedSpatialStructure instance relationship. It is possible to extract only the necessary elements by utilizing this relationship. Specifically, this becomes a partial graph that includes the specific IfcBuildingStorey instance, the physical element instance connected to the IfcRelContainedSpatialStructure instance, and all instances connected at the relationship of IfcRelContainedSpatialStructure.

3.3 Conversion to Property Graph Model

The IFC product model is converted into a property graph model, giving consideration to the partial extraction method described above. In order partially to extract shape data, it

is necessary to judge whether each instance is an instance of an IfcProduct-derived class, so it is necessary to include class succession information. Also, partial extraction of data of a specific location uses the relationship of IfcRelContainedSpatialStructure, so it must include relationship information for the constituent elements.

There is a difference in the types that can be used with EXPRESS (ISO 2004), which is the schema definition language of the IFC schema, and the property graph model. Therefor it is necessary to determine the method of expression of the type. No standard has been defined like EXPRESS for the property graph model, so the reality is that the specifications of the type depend on the product. In this research, a type conversion rule was defined in line with the graph DBMS product called Neo4j.

The following describes the conversion procedures from the IFC product model considered above to the property graph model.

1. Create a node that corresponds to an instance other than the instance that expresses the relationship. Also, store the class name of the instance in the node label.
2. If the instance attribute is a simple data type, or enumerated data, store the attribute value by creating a property with the same name as the attribute name, in the node.
3. If the instance attribute is an aggregate data type, the aggregate element data type is a simple data type, or enumerated data type, store the aggregate element arrangement in the node by creating a property with the same name as the attribute name.
4. If the instance attribute is an entity data type, or an aggregate data type of the entity data type, the nodes of the instances that correspond to the attributes are connected at an oriented edge toward the node.
5. Created in procedures 1-3. Then, store the class succession information in the property whose name is superClass.
6. For instances that indicate the relationship, connect from the nodes that correspond to the related source instance at an oriented edge on the node that corresponds to the related destination instance. Also, store the class name of the instance that indicates the relationship in the property whose name is "class".

With the property graph model generated with the rules above, the succession information of the class is held in the superClass property in each node. If the class that is the source is the same, the succession information is also the same, so retaining succession information in each node is redundant, but while the graph DBMS accesses localized data very quickly, that accessing speed slows down for data that is widely dispersed. For that reason, frequently referenced information was localized. Figure 3 shows an example conversion from an IFC product model to a property graph model.

4 VERIFICATION

4.1 Verification Method

The authors verified that the conversion method and partial extraction method described in Section 3 can be adopted for an actual product model. An IFC product model of a five-storey building (Table 1) was used as the target data.

Also, query for partial extraction used the figure 4. This is the meaning of the executed query, but the first three lines specify the building and the floor to extract from various building data stored in the Neq4j DBMS, and further indicates extracting only the nodes that include the shapes. Specify up to 5F of the building using the project name (that indicates the entire building) "LUC-01," and the floor name "Mod-Floor-5," and further, by extracting only instances of the derived class of IfcProduct, instances that do not include shapes are excluded. The next two lines specify the shape data to extract in specific members (stairs), and means to extract the faces and points that constitute that. The last line indicates output data. This outputs the ID for identifying faces, and coordinates (x, y, and z) of each point.

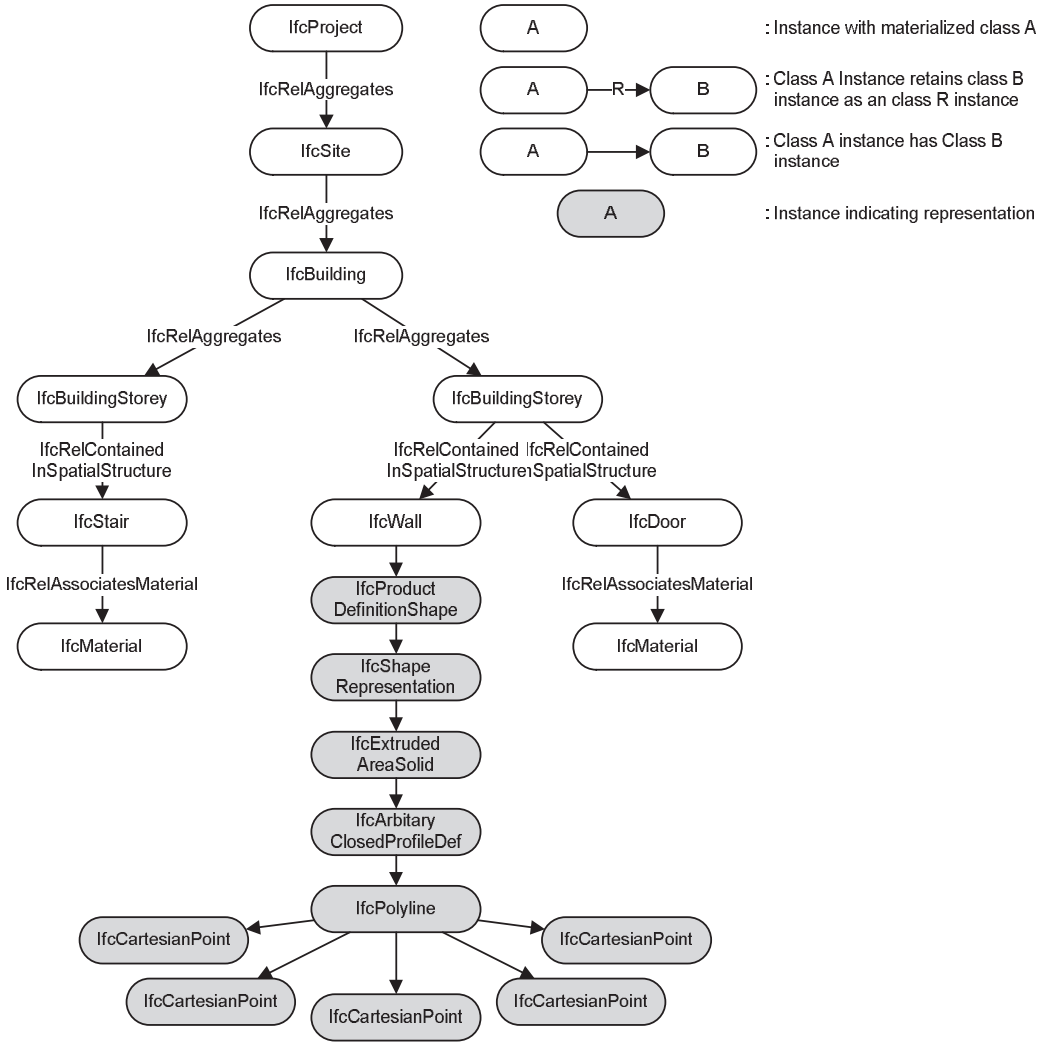

Figure 3: Expression in Property Graph Model of IFC Product Model

Table 1: Measurement of Target Product Model

Item	Quantity
file size	51M bytes
number of instances	1,003,212
max depth	19
average depth	11.17

```
MATCH (p:IfcProject)-[*]->(f:IfcBuildingStorey)-[*]->(s)
WHERE p.name="LUC-01" AND f.name="Mod-Floor-5" AND "IfcProduct" IN s.superClass
WITH s MATCH (s)-[*]->(l:IfcPolyLoop)-->(p:IfcCartesianPoint)
WHERE s.shapeType="STAIR"
RETURN id(l), p.coordinates[0], p.coordinates[1], p.coordinates[2]
```

Figure 4: Query for Partial Extraction

4.2 Results and Discussions

Table 2 shows measurement values of the partial extraction processing results; Figure 5 shows the results displayed in the viewer by converting the obtain shape data into the obj format (Wikipedia 2016). These show that correct partial extraction was possible.

Figure 2 shows that the IFC product model is a hierarchical structure. However, because this is a structure in which the most significant element includes the least significant element, it is possible to obtain up to the coordinate points (IfcCartesianPoint) of the terminal by following the least significant elements from the project (IfcProject) that is the most significant element. Specifically, the extraction process in the IFC product model narrows down the data to target in the order of (1) all buildings, (2) a specific building, (3) a specific floor, (4) specific types of members, and (5) specific types of data.

The extraction process in the graph database is a process that scans all nodes that are targeted, and considers targeted number of nodes as n, thereby increasing processing speed at normal $O(n)$. Therefore, processing to calculate various building information is weak, but with processing to narrow down the targeted nodes as described above, the graph scanning process becomes localized so it operates at high speed.

5 Conclusions

It is clear that when IFC product models are stored in a relational DBMS, data acquisition processing performance degrades as the hierarchical structure of the target becomes deeper. Conversely, the graph DBMS has no performance degradation. Also, it is clear that the property graph model adopted for the graph DBMS has good compatibility with the IFC schema design policy that codes in detail the relationship of the constituent elements, and can be compactly turned into a model. Furthermore, consideration was given to a conversion method for storing the IFC product model in the graph DBMS. It was verified that it is simple to extract only specific data from the DBMS that stores models after conversion by using only a feature (query) equipped in the Neo4j DBMS, without having to use complex programs. That fact suggests that a BIM inofrmation-sharing system can be developed easily using the graph DBMS, unlike existing systems.

Table 2: Partial Extraction Results

Item	Value
Number of Obtained Polygons	13,440
Number of Obtained Points	40,320
Execution Time	4,242ms

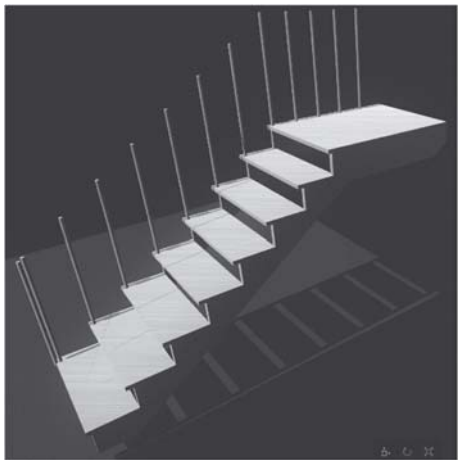

Figure 5: Partially Extracted Stairs for 5F

This research focused on the partial extraction processing method itself which uses a graph DBMS, so its processing speed was not addressed. The proven extraction process ran at a speed that is within a practical range, but that speed is not considered to be adequately fast (Table 2). Therefore, attaining faster extraction processing speeds are future issues to address in the graph DBMS.

6 References

Beetz, J., van Berlo, L., de Laat, R., van den Helm, P. (2010). BIMSERVER.ORG – An Open Source IFC Model Server, *Proceedings of the CIB W78 2010: 27th International Conference*.

Das, M., Cheng, J. and Kumar, S. (2015). *Social BIMCloud: a distributed cloud-based BIM platform for object-based lifecycle information exchange*.

ISO (2004). Product data representation and exchange -- Part 11: Description methods: The EXPRESS language reference manual

Microsoft (n.d.). *SQL Server 2008 R2 Editions Overview*. Available at: https://assets.microsoft.com/en-us/SQLServer2008R2EditionsDatasheet_1.pdf [Accessed 1 Nov. 2016]

Neo Technology (n.d.). Neo4j: *The Words's Leading Graph*, Available at: http://neo4j.com/ [Accessed 1 Nov. 2016]

Robinson, I., Webber, J. and Eifrem, E. (2013). Graph Databases New Opportunities for Connected Data:O'Reily. ISBN: 978-1-449-35626-2.

Wikipedia (2016). *Wavefront .obj file*. Available at: https://en.wikipedia.org/wiki/Wavefront_.obj_file#cite_ref-3 [Accessed 1 Nov. 2016]