Linked Data for Smart Homes: Comparing RDF and Labeled Property Graphs

Alex Johannes Albertus Donkers[1][0000-0002-8809-3277], Dujuan Yang[1][0000-0003-3124-6604] and Nico Baken[1]

[1] Eindhoven University of Technology, Groene Loper 6, 5612AZ Eindhoven, The Netherlands
a.j.a.donkers@tue.nl

Abstract. The need to integrate siloed data in the built environment led to a gaining interest in semantic web technologies in the Architecture, Engineering and Construction (AEC) sector. Especially for smart home developments, the integration of information about the building, users and Internet of Things (IoT) devices could be valuable. The Resource Description Framework (RDF) is the standard model for the semantic web, however, labeled property graphs (LPG) also proved to be effective in linking data. This research used the Open Smart Home Dataset and a dataset representing a kitchen to compare the two graph models both qualitatively and quantitatively. The comparison shows that native labeled property graphs are less complex and outperform the atomic RDF in complex graph traversals. However, RDF shows qualitative advantages for multi-domain and multi-stakeholder environments, such as the use of ontologies and HTTP URIs, making it a more stable interoperability format.

Keywords: RDF, LPG, Neo4j, Smart Home

1 Introduction

There is a lot of potential in using IoT data from our built environment. The integration of IoT data could lead to a range of new insights in building operations, energy consumption, circularity and more. However, data captured in the built environment is often stored in organizational silos, which could not easily interconnect due to technological, managerial and governance implications. Integration of knowledge in the built environment is vital since the sector is characterized by the involvement of multiple stakeholders with their own domain knowledge, working in different lifecycle phases.

The knowledge sharing capabilities of Building Information Modeling (BIM) remain limited to the integration of files. Semantic web technologies, first mentioned by Berners-Lee [9], aim to create structured connections between different sources of information, in order to enrich the information exchange between these sources. This enables the integration of heterogenous building datasets from different stakeholders into a web of data. Therefore, linking building data through semantic web technologies (SWT) is increasingly used in the architecture, engineering and construction (AEC) industry [38]. Early initiatives by, amongst others, Beetz et al. [8] led to the development of SWT in the AEC sector. They discussed the development of an ontology for

the AEC industry – ifcOWL – capturing IFC data in an RDF graph to allow linking building data to other information. Many extensions of ifcOWL have been proposed, as described by Pauwels et al. [38]. Pauwels and Roxin [37] and Rasmussen et al. [42] developed simplified ontologies for buildings, describing the core concepts of a building. This resulted in the Building Topology Ontology (BOT), which could be extended with domain knowledge. The development of IFC-to-RDF converters, such as the converter by Oraskari [11] (using the BOT ontology) and the converter by Pauwels (using ifcOWL) [39] eased the process of using linked data technologies. Following these developments, many extensions to the construction ontologies have been made. These often reflect the different stakeholders in the construction process and their domain knowledge [38].

There is a recent paradigm shift regarding the provision of health and comfort in buildings, illustrated by amongst others the WELL Certification for healthy buildings. The stronger focus on indoor environmental concepts such as light, thermal comfort, air and sound led to IoT-developments in buildings, in order to create 'Smart Home' concepts. Marikyan et al. [31] performed an extensive review and mentioned the integration of devices in smart homes as a future research avenue. They mentioned several functions of smart homes: daily routine automation, remote home management, environmental services, smart leisure, health and lifestyle monitoring, remote health interaction and therapy, supporting patients with hearing issues, mobility issues, socialization, visual disabilities or home rehabilitation and giving suggestions. The technologies realizing these functions often combine multiple physical devices, IoT data and a location, and therefore fit the linked data approach.

The increasing research into linked data for AEC led to the integration of building data with other data sources for typical smart home cases, such as integrating IoT data [5, 30, 50]. Some initiatives focused on specific smart home cases, such as health monitoring [40, 54], social media integration [15], indoor environmental quality [16, 17, 35], energy efficiency [22, 43, 48], activity recognition [14], home automation [30] and building monitoring [1, 21]. These initiatives either create their own fit-for-purpose ontologies, or use existing ontologies related to smart homes. To be more specific, the SSN and SOSA [25] ontologies are able to capture sensors and their observations. DogOnt [12] aims to link smart, electronical devices to spaces. ThinkHome [43] is a knowledge representation for smart homes. SEAS [47] is able to connect physical systems with IoT measurements, and therefore link observations to a physical object. The SAREF ontology [45] describes devices in smart home environments. The BOnSAI [49] and Brick [13] ontologies extensively describe hardware in smart homes, coupled to the indoor context. A full overview has been given by Pauwels et al. [38] and includes initiatives linking building data to detailed product information, laws and regulations, and geometric and geographic information.

The semantic web has been dominated by the resource description framework (RDF) as W3C's standard model. Many ontologies and vocabularies have been openly published (the linked open vocabularies cloud currently consist out of 697 vocabularies) and developments are carried out in the W3C Linked Building Data Community. Simultaneously, the concept of labeled property graphs (LPG), often using the native graph database Neo4j, is getting more attention lately. The model has been used in multiple

AEC-related use-cases and has shown that typical characteristics of LPGs could be beneficial to linked data models for smart homes and cities, such as using relationship labels [23, 36], fast and easy graph search [24], scalability [34] and performing complex graph algorithms [36]. While different graph models proved to be useful in multiple use-cases, there is a lack of empirical comparison of these graph models for linking data in the AEC industry.

This paper reviews different graph models by their core characteristics and their implementation in the smart home domain in Section 2. After selecting the two most mature models, the paper discusses the methodology of this research in Section 3. The differences between the two graph models (RDF and LPG) for integrating smart home data both qualitatively and quantitatively are presented in section 4 which is followed by a conclusion in section 5.

2 Graph models and their implementation

Rodriguez & Neubauer [44] collected different types of graph models. Based on their research, we compared their characteristics in Table 1. This section compares the maturity of these graph models based on state-of-the-art research in the smart home domain.

Table 1. Graph models and their characteristics.

Graph model	Directed edges	Labels	Attributes	URI	>1 edges between nodes	Weights	Edges join >2 nodes
Undirected graph		○					
Simple graph	○						
Multi-graph	●				●		
Halve-edge graph		○					
Labeled graph	●	●					
Weighted graph	●	○	●			●	
Hypergraph	○	○	○		○	○	○
Property graph	●		●		○	○	
Labeled property graph	●	●	●		○	○	
RDF	●	●		●	○		
RDF*	●	●	○	●	○	○	

● = Always, ○ = Possibly

Undirected graphs, simple graphs, multi-graphs, halve-edge graphs, labeled graphs and weighted graphs could all be considered to be simplified LPGs [44]. Although in some use-cases they might be useful, we do not consider these models to be real competitors to the other graph models.

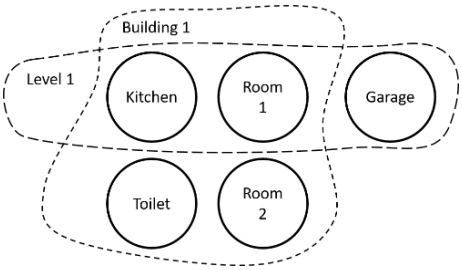


Fig. 1. An example of linked building data using a hypergraph model

The hypergraph model is used by the Grakn knowledge graph, which allows edges to join multiple nodes (Fig. 1). The hypergraph model allows users to model new types of relationships (in Grakn referred to as hyper-relationships) such as N-ary and nested relationships [3]. Although limited hypergraph use-cases for smart homes have been found, the theory is deployed in some IoT cases. Qu, Tao and Yuan [41] created a blockchain architecture using hypergraphs, while Jung et al. [28] modeled IoT devices in smart homes using the hypergraph model. The theory has also been used for machine learning operations using the MySQL World database, consisting of country-scale information [32] and to create a design engineering assistant for space missions in [10].

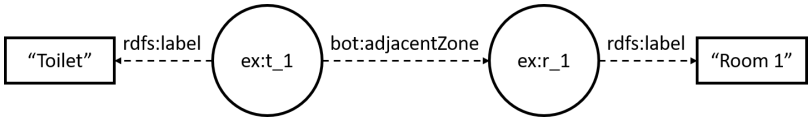


Fig. 2. An example of linked building data using an RDF model

In RDF, data is linked via a subject-predicate-object structure (Fig. 2). The subject is a node, the predicate is an edge and the object is either another node or a literal. Multiple triples could be mentally modeled as a graph of data consisting of nodes and edges, named by a Unique Resource Identifier (URI) which typically is an HTTP URI. A unique asset of RDF is that the HTTP URIs can be published on the world wide web, and therefore be used by others. Simultaneously, a user can easily use other RDF triples to link to its own graph database, resulting in rich datasets with information from many sources. SPARQL has been adopted as the standard query language for RDF and is also based on triple patterns. It's a graph query language which retrieves information from a graph based on pattern matching. Being a global standard, RDF is the most frequently used graph model in the smart home domain and has been used in cases related to energy [22], home automation [5, 30], health [54], activity-recognition [14] and IoT integration [5, 50].

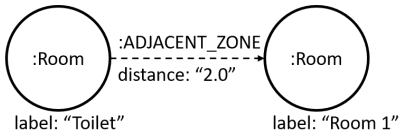


Fig. 3. An example of linked building data using an LPG model

A popular graph model next to RDF is the labeled property graph, natively used by Neo4j. Different from RDF, LPGs can carry properties directly within their nodes and relationships (Fig. 3). The internal structure of nodes and relationships is described by key-value pairs. LPGs are typically node-centric, different from the edge-centric RDF triples [2]. Cypher is the main query language which is used in LPGs. Other graph query languages are Oracle's PGQL, TigerGraph's GSQL and LDBC's GQL. As the LPGs do not use a schema language, Cypher uses no prefixes and is merely a combination of Cypher keywords. The LPG model already proved to be useful for different smart buildings and city use-cases. It has been used for cases related to smart city infrastructure [24], citizen recommendation services [36], energy smart buildings [34] and geospatial data [23].

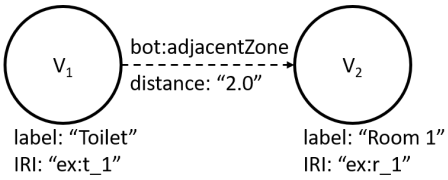


Fig. 4. An example of linked building data using an RDF* model

The RDF* model aims to combine the strengths of RDF and LPG. There is a proposition for extending the RDF model with the possibility to add attributes, named RDF* [26]. RDF* enables adding metadata to statements through the notion of nested triples, based on reification [4] (Fig.4). However, RDF* is still in a research phase.

While both the hypergraph, LPG and RDF* models are considered to be suitable competitors to RDF, LPG is the most mature in terms of community, software development and state-of-the-art use-cases to the best of our knowledge [3, 19]. Therefore, this research will focus on comparing LPG and RDF.

3 Methodology

The evaluation consists of both objective and subjective comparisons between the two graph models, based on two case studies. The subjective (qualitative) comparison compares the use of both models for smart homes based on their fundamental differences, differences in relationships, inference, the use of ontologies, their query languages, interoperability and other measures. The objective (quantitative) comparison compares measurable differences between the models in their operating phase, including query execution time and storage, but also graph complexity (consisting of node and edge counts and graph density). To compare, the Open Smart Home dataset [46] and a dataset representing a kitchen have been used. The details of the comparison are explained in the following sub-sections.

3.1 Qualitative Comparison

A qualitative comparison is conducted based on the evaluation measures used in previous research. Insights from a literature review have been projected on possible smart home use-cases to compare the graph models. Following this review, we compare fundamental differences [4, 18], query languages [2], ontologies, relationships [7], inference [3] and other measures like interoperability, usability, support and security [2, 3].

3.2 Quantitative Comparison

The major focus of the quantitative comparison is comparing the structural differences of the graph models and differs from a lot of previous research which mainly focused on comparing graph database systems ([20, 27, 52]). A comparison of two graphs will be performed using two datasets. First, the Open Smart Home Dataset [46], which is openly available from GitHub and comes in RDF Turtle format, will be used. The dataset represents an IFC file of a flat with different rooms, of which most of them are equipped with different sensors (measuring temperature, brightness and humidity). It also contains different RDF representations, of which the one using the BOT ontology has been used for this research. An LPG has been created using the NSMNTX plugin [6], which is used to import and export RDF data into Neo4j as an LPG. NSMNTX transforms triples using an -n-K-V and -e-K-V schema (as explained in [18]), where *n* (node), *e* (edge), and *K* (key), are URIs and *V* (value) is a literal. More development is necessary to be able to transform more complex topologies.

Next to the Open Smart Home data, two datasets representing a kitchen have been created using the two graph models. One of them is a typical RDF turtle file based on the same structure as the Open Smart Home data, while the other is a typical LPG following no schema language. For comparison, both datasets have been imported using NSMTX in Neo4j. They are stored as local Neo4j graphs to compare the performance of the different initial structures of the data. These quantitative comparisons include measuring the query execution time based on four different queries and 880 runs, required storage space and different measures for graph complexity. This includes counting the nodes and edges, but also the graph density, as calculated in equation 1:

$$\rho_{graph} = \frac{|Edges|}{|Nodes|*|(Nodes-1)|} \quad (1)$$

4 Results

4.1 Qualitative Comparison

Fundamentals. RDF has been designed as a framework for publishing and exchanging data amongst a large group of stakeholders in a structured format. LPG has been developed mainly for using the data, with the purpose of storing and querying it as efficient as possible [7]. RDF and LPG have two fundamental differences:

1. RDF nodes and edges can't hold properties; LPG nodes and relationships can;
2. RDF is often index- and schema-based; LPG is not.

Due to RDFs lack of internal structure, properties of nodes can only be described by adding new nodes or literals. This results in a rather atomic decomposition of entities. Compared to RDF, LPGs are more compact, as entities' properties are stored within the nodes or edges. This could lead to a difference in nodes and triples of an order of magnitude [7].

Relationships. LPGs allow relationships to carry properties. Based on the mentioned smart home functions, there is potential in describing the relationships between people, IoT devices and the physical built environment. LPG's key-value pairs could be used to describe preferences, rules, restrictions and other information. In RDF, nodes could carry properties by adding new nodes and relationships, but edges are unable to carry such information without complex workarounds (shown in [18]). An advantage of putting attributes on edges is that it allows to easily create temporal relationships and weighted relationships [33]. Temporal RDF constructs are more cumbersome. Weighted relationships can be used for network analysis algorithms.

Since RDF uses URIs to define nodes and relationships, it cannot create multiple instances of the same relationship. For example, one cannot state the following in RDF:

```
:Sensor :measured :Temperature
:Sensor :measured :Temperature
```

It wouldn't count the fact that the sensor measured the temperature two times. To do so, the RDF file needs a workaround, such as using observations:

```
:Sensor :measured :Observation1
:Observation1 :hasProperty :Temperature
:Sensor :measured :Observation2
:Observation2 :hasProperty :Temperature
```

On the other hand, LPG cannot handle multivalued properties [4]. The following construct would not be possible in LPG:

```
CREATE (building {restrict.: "Bob", restrict.: "Lisa"})
```

The query would only show the restriction of Lisa and neglect the restriction of Bob. Again, there's workaround, such as using an array:

```
CREATE (building {restrict.: ["Bob", "Lisa"]})
```

It can be concluded that, based on the relationships, the best graph model depends on the use-case. There's potential in developing easier workarounds for both models.

Ontologies. Different from LPGs, RDF stores typically support ontology modeling languages like RDFS and OWL2 RL [18]. Ontologies have two main purposes. First, they are used as an interoperability framework; different stakeholders use the same framework to model their data. Second, ontologies store domain knowledge, allowing machines to do inferencing on data, in order to better understand the data or to derive new insights. They have both advantages and disadvantages which will be discussed below.

First, ontologies allow machines for deeper reasoning of data, as they have built-in relationships between concepts. This reasoning is very similar to human reasoning.

Many of this reasoning is done through inferencing: creating new knowledge from existing information [3, 4]. Secondly, ontologies in the semantic web stack are easily extendable. One can mix multiple ontologies, create new ones and extend existing ones to create fit-for-purpose schema language. A third advantage of ontologies is that they structure domain knowledge. In the built environment, many stakeholders from different domains work together, all with their own rules and knowledge. By using domain-specific ontologies, machines could re-use this knowledge without the need to specify them every single time. Finally, ontologies do not only allow knowledge sharing across different domains, they also stimulate the re-use of knowledge amongst stakeholders within domains. When modeling many similar objects with different owners, schema language might be desirable to allow machines to deal with the graphs similarly, for example by using the same software or queries. Especially in the built environment, with so many buildings, devices and other entities to be modeled, all with their own owners, basic schema language can be used to integrate data better.

A disadvantage of using ontologies is that they limit the freedom of describing entities. As knowledge is formally defined a priori, anyone who wants to describe an entity using ontologies is somewhat bounded by this knowledge. Another disadvantage is that ontologies need a certain level of agreement amongst stakeholders. If multiple stakeholders use different mixtures of ontologies, many of the advantages of ontologies fade away. Possibly, central bodies should advise the different stakeholders about standards.

Query language. SPARQL and Cypher are both declarative, SQL-like query languages. Although the authors felt that Cypher is more intuitive and simpler to use, a comparison by Keen [29] based on multiple queries shows that the complexity of the queries in both languages is in fact very similar. As SPARQL is standardized, RDF triples could be queried by different applications flawlessly [2]. SPARQL can also query results from multiple databases combined, directly [33]. LPGs still use multiple query languages, which limits interoperability amongst users and software [2, 4]. However, there is an industry push to create global standard graph query language named GQL, strongly influenced by Cypher. For now, SPARQL seems to be the most stable and most portable option.

Other measures. Being a global standard, RDF is supported by W3C and many database vendors, offering mature software tools [18]. Support for LPG is more fragmented [2]. Its development is supported by the Neo4j community (7000 users) developing the model through Neo4j Labs. However, the interest in LPG is growing. A google trend report comparing Neo4j and RDF (since 2009) shows a relative increase in popularity for Neo4j.

The security of both RDF and LPG is vendor-dependent, as different databases have different security measures. Data security is important for smart homes and preferably, owners would be able to choose which clients can traverse over which data. Since RDF uses HTTP URIs, the domain owner can implement security restrictions to the URIs [33]. Neo4j has a role-based system for user roles, allowing admins to determine through which nodes in a graph a client can traverse. It also includes options for white-listing and restrictions for external APIs.

As RDF is intentionally designed as a model to share data on the web, the model has certain interoperability perks. In RDF all nodes and edges are globally unique, and therefore discoverable by others. Next to that, the ontologies enable sharing knowledge with many stakeholders. In LPG, identifiers are local which limits the possibility to integrate data across multiple stakeholders and systems [33]. Another interoperability perk which RDF has is provenance: a description of the origin of the data. This could be done either by inference or through ontologies (such as www.w3.org/ns/prov#) [33].

LPGs generally use a closed world assumption, while RDF schema language concepts often rely on an open world assumption [53]. This opens many possibilities for inference and generally fits the concept of the semantic web better [38].

4.2 Quantitative Comparison

The RDF graph and LPG graph highly differ in complexity as the RDF graph is more atomic. This could be reflected by counting the nodes, edges and the graph density. Table 2 shows the differences for the open smart home data.

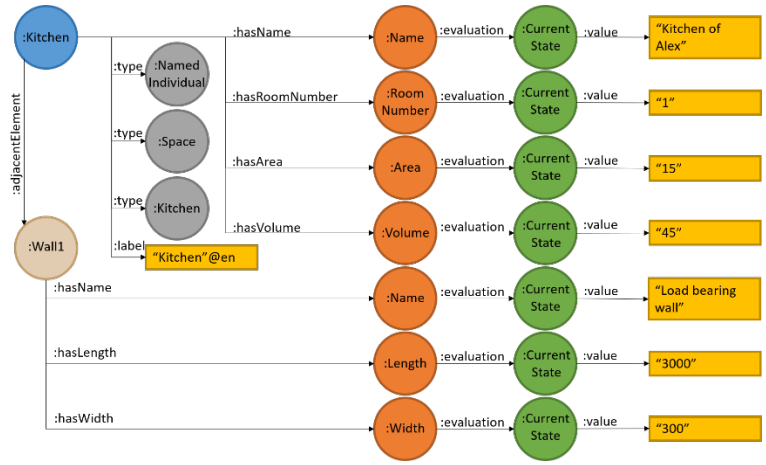
Table 2. Quantitative differences between RDF and LPG graphs for open smart home.

Heading level	RDF	LPG
Nodes	719	475
Edges	1332	507
Graph density	0,0026	0,0023

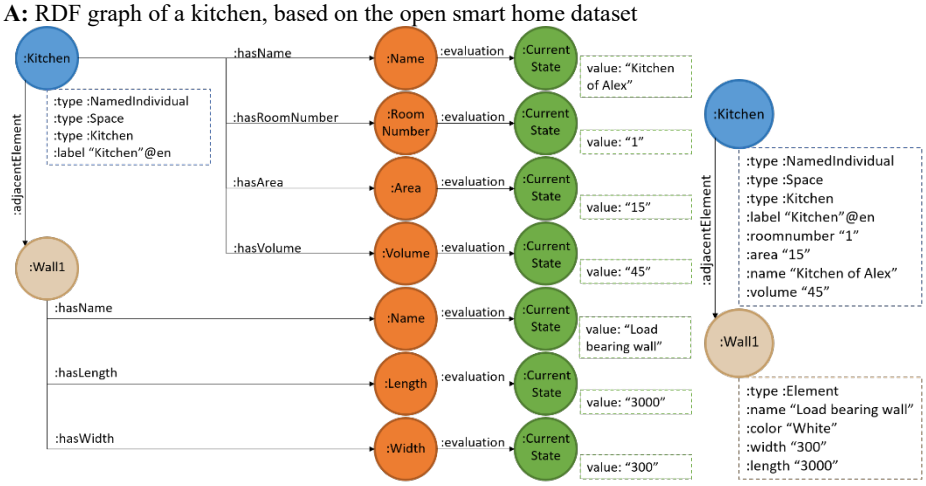
The difference in nodes and edges is clear. From the RDF graph, 549 out of the 719 nodes were actual HTTP URIs while the others were literals. These literals, plus some of the HTTP URIs, have been added to the LPG nodes either as key-value pairs or as node labels. However, the graph density of both graphs is not so different. Bigger differences could be seen when more direct information about objects is added (Table 3 and Figure 5). A typical RDF graph (based on the BOT ontology) has been converted to an LPG using the NSMTX plugin of Neo4j. Similar information has been described in a custom LPG graph. It is clearly visible how much simpler the LPG graphs are compared to the RDF graph. However, [18] argues that the cardinalities strongly depend on the transformation schema.

Table 3. Quantitative differences between RDF and LPG graphs for a kitchen.

Heading level	RDF	LPG (using NSMTX)	LPG (custom)
Nodes	27	16	2
Edges	26	15	1
Graph density	0,037	0,063	0,5



A: RDF graph of a kitchen, based on the open smart home dataset



B: LPG graph using the NSMTX RDF import in Neo4j **C:** Custom LPG graph

Fig. 5. Comparison between RDF and LPG graphs

Clear differences in query execution time could also be seen between the LPG using a NSMTX RDF import and the custom LPG. Different queries have been performed in Neo4j to measure the difference in execution time (Listing 1). As Neo4j maps data files and query execution plans to the cache after querying, the first queries after connecting to the server were found to be considerably slower [51]. Therefore, both the average of the first runs as well as the average after caching (100 runs) are compared in Table 4. More complex queries were easier to create for the custom LPG graph, as the query depth is lower, and therefore more intuitive. These results correspond to earlier findings [2, 7], stating that the cost of traversing edges in an RDF graph is logarithmic, while LPGs are designed for fast traversing. De Abreu et al. [20] found RDF databases to be sensitive to graph density.

Listing 1. Cypher queries to compare two graphs

```
1. MATCH (n) RETURN n
2. MATCH (n)-[r]->(m) RETURN n,r,m
3. MATCH (n)-[r:ns0__AdjacentElement]->(m) RETURN n,r,m
4a.MATCH (n)-[:ns3__dimensionsLength]->()-[]->(l),
  (n)-[:ns3__dimensionsWidth]->()-[]->(w),
  (n)-[:ns3__identityDataName]->()-[]->(s)
  RETURN n.uri, l.sch__value, w.sch__value, s.sch__value
4b.MATCH (n)-[]->(m)
  RETURN m.uri, m.ns2__name,m.ns2__width,m.ns2__length
```

Table 4. Comparing queries for NSMTX RDF import versus custom LPG graphs.

Query	LPG (NSMTX) First 10 run avg.	LPG (NSMTX) 100 runs avg.	Custom LPG First 10 run avg.	Custom LPG 100 runs avg.
1	23.8 ms	2.6 ms	13.8 ms	1.3 ms
2	32.5 ms	4.4 ms	16.8 ms	1.6 ms
3	27.8 ms	1.8 ms	18.0 ms	1.6 ms
4a, 4b	23.7 ms	1.7 ms	11.5 ms	1.2 ms

De Abreu et al. [20] found that RDF stores outperformed LPG databases for graph creation and simple queries, but fell behind during more complex queries. Jouili and Vansteenberghe [27] found that Neo4j outmatched other databases on graph traversals. However, this largely depends on the topology of the graph [18]. A final difference could be found in the storage space of both graphs, with the RDF graph using 7kb storage compared to 2kb for the custom LPG. This makes sense, since the custom LPG stores the data in a less atomic way and therefore needs less characters. However, Das et al. [18] argues that RDF stores have no limit in storage size, while LPGs are limited by their graph database software, making RDFs more likely to serve as backend storage.

5 Conclusion and Discussion

The increasing availability of IoT data in the built environment asks for new methods of data integration. Therefore, linked data principles are gaining attention. This paper compared two graph models for linked data: the established RDF and LPG, which is growing in popularity. Both a qualitative and quantitative comparison have been conducted. Quantitatively, the native LPG outscored the RDF in all performance measures. For real-time operations, we conclude that LPG performs best.

However, qualitatively, RDF might be favorable in some scenarios. RDF exceeds LPG in multi-stakeholder environments, as it is able to share domain knowledge amongst many stakeholders (both within and without the domain) and do inference based on this knowledge. Considering the proposed functionalities of smart homes and the need to integrate many different types of data to build applications, RDF is considered the most stable model.

We consider the small dataset and the limited amount of queries a limitation, and therefore propose future research which should compare larger datasets using queries based on different use-cases (related to indoor environmental quality, energy efficiency, maintenance and geometric queries) in different query languages. These datasets will also contain more complex topologies to test various transformation schemas.

It is necessary to closely follow the developments of hybrid variants of RDF and LPG, such as RDF*, but also the interoperability of RDF and LPG, such as presented in [4] and [18], but also in the recent NSMTNX developments [6]. These future developments might combine the strengths of both graph models.

References

1. Ahvar, S., Santos, G., Tamani, N., Istasse, B., Praça, I., Brun, P. E., ... , Crespi, N.: Ontology-based model for trusted critical site supervision in FUSE-IT. In: 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), pp. 313-315, IEEE (2017).
2. Alocci, D., Mariethoz, J., Horlacher, O., Bolleman, J.T., Campbell, M.P., Lisacek, F.: Property Graph vs RDF triple store: A comparison on glycan substructure search. In: PLoS One, vol. 10(12) (2015), https://doi.org/10.1371/journal.pone.0144578
3. Altinok, D.: Neo4j vs GRAKN Part I: Basics (2020), https://towardsdatascience.com/neo4j-vs-grakn-part-i-basics-f2fe3511ce88
4. Angles, R., Thakkar, H., & Tomaszuk, D.: RDF and property graphs interoperability: Status and issues. In: Proceedings of the 13th Alberto Mendelzon International Workshop on Foundations of Data Management, Asunción, Paraguay (2019).
5. Balakrishna, S., Thirumaran, M.: Towards an optimized semantic interoperability framework for IoT-based smart home applications. In: Intelligent Systems Reference Library, vol. 154 (2019)
6. Barrasa, J.: Neosemantics (2016), https://github.com/neo4j-labs/neosemantics
7. Barrasa, J.: RDF Triple Stores vs. Labeled Property Graphs: What's the Difference? (2017), https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/
8. Beetz, J., Van Leeuwen, J., De Vries, B.: IfcOWL: A case of transforming EXPRESS schemas into ontologies. AI EDAM, vol. 23(1), pp. 89-101 (2009).
9. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web (2001)
10. Berquand, A., Murdaca, F., Riccardi, A., Soares, T., Generé, S., Brauer, N., Kumar, K.: Artificial intelligence for the early design phases of space missions. In: 2019 IEEE Aerospace Conference, pp. 1-20. IEEE. (2019).
11. Bonduel, M., Oraskari, J., Pauwels, P., Vergauwen, M., Klein, R.: The IFC to linked building data converter - Current status. In: CEUR Workshop Proceedings, vol. 2159, pp. 34-43 (2018)
12. Bonino, D., & Corno, F.: Dogont-ontology modeling for intelligent domotic environments. In: International Semantic Web Conference, pp. 790-803. Springer (2008).
13. Brick Schema. http://brickschema.org/ontology/
14. Chen, L., Nugent, C.D., Wang, H.: A knowledge-driven approach to activity recognition in smart homes. In: IEEE Transactions on Knowledge and Data Engineering, vol. 24(6), pp. 961-974 (2012), https://doi.org/10.1109/TKDE.2011.51
15. Corry, E., O'Donnell, J., Curry, E., Coakley, D., Pauwels, P., Keane, M.: Using semantic web technologies to access soft AEC data. In: Advanced Engineering Informatics, vol. 28(4), pp. 370-380 (2014)

16. Corry, E., Pauwels, P., Hu, S., Keane, M., & O'Donnell, J.: A performance assessment ontology for the environmental and energy management of buildings. In: Automation in Construction, vol. 57, pp. 249-259 (2015).
17. Curry, E., O'Donnell, J., Corry, E., Hasan, S., Keane, M., & O'Riain, S.: Linking building data in the cloud: Integrating cross-domain building data using linked data. In: Advanced Engineering Informatics, vol. 27(2), pp. 206-219 (2013).
18. Das, S., Srinivasan, J., Perry, M., Chong, E. I., & Banerjee, J.: A Tale of Two Graphs: Property Graphs as RDF in Oracle. In: EDBT, pp. 762-773 (2014).
19. DB-Engines Ranking. https://db-engines.com/en/ranking
20. De Abreu, D., Flores, A., Palma, G., Pestana, V., Pinero, J., Queipo, J., Sánchez, J., Vidal, M.E.: Choosing between graph databases and RDF engines for consuming and mining linked data. In: COLD'13: Proceedings of the Fourth International Conference on Consuming Linked Data, vol. 1034, pp. 37-49 (2013).
21. Dibley, M., Li, H., Rezgui, Y., & Miles, J.: An ontology framework for intelligent sensor-based building monitoring. In: Automation in Construction, vol. 28, pp. 1-14 (2012).
22. Fensel, A., Tomic, S., Kumar, V., Stefanovic, M., Aleshin, S. V., Novikov, D.O.: SESAME-S: Semantic smart home system for energy efficiency. In: Informatik-Spektrum, vol. 36, pp. 46-57 (2013). https://doi.org/10.1007/s00287-012-0665-9
23. Ferreira, D.: Using Neo4J geospatial data storage and integration (2014).
24. Gorawski, M., Grochla, K.: Performance Tests of Smart City IoT Data Repositories for Universal Linear Infrastructure Data and Graph Databases. In: SN Computer Science, vol. 1 (2020). https://doi.org/10.1007/s42979-019-0031-y
25. Haller, A., Janowicz, K., Cox, S. J., Lefrançois, M., Taylor, K., Le Phuoc, D., ... & Stadler, C.: The SOSA/SSN ontology: a joint WeC and OGC standard specifying the semantics of sensors observations actuation and sampling. In: Semantic Web, vol. 1, pp. 1-19 (2018).
26. Hartig, O.: Foundations of RDF∗ and SPARQL∗ :(An alternative approach to statement-level metadata in RDF). In: AMW 2017 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web, Montevideo, Uruguay, vol. 1912 (2017)
27. Jouili, S., Vansteenberghe, V.: An empirical comparison of graph databases. In: 2013 International Conference on Social Computing, pp. 708-715 (2013)
28. Jung, J., Chun, S., Lee, K. H.: Hypergraph-based overlay network model for the Internet of Things. In: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), pp. 104-109. IEEE (2015).
29. Keen, A.: SPARQL and Cypher Cheat Sheet (2018). https://dzone.com/articles/sparql-and-cypher.
30. Machorro-Cano, I., Paredes-Valverde, M.A., Alor-Hernandez, G., del Pilar Salas-Zárate, M., Segura-Ozuna, M.G., Sánchez-Cervantes, J.L.: PESSHIoT: Smart Platform for Monitoring and Controlling Smart Home Devices and Sensors. In: International Conference on Technologies and Innovation, pp. 137-150 (2019)
31. Marikyan, D., Papagiannidis, S., Alamanos, E.: A systematic review of the smart home literature: A user perspective. In: Technological Forecasting and Social Change, vol. 138, pp. 139-154 (2019). https://doi.org/10.1016/j.techfore.2018.08.015
32. Masood, A., Hashmi, A.: AIOps: Predictive Analytics & Machine Learning in Operations. In: Cognitive Computing Recipes, pp. 359-382. Apress, Berkeley, CA (2019).
33. McComb, D.: Property Graphs: Training Wheels on the way to Knowledge Graphs (2019). https://www.semanticarts.com/property-graphs-training-wheels-on-the-way-to-knowledge-graphs.
34. Nizamic, F., Nguyen, T.A., Lazovik, A., Aiello, M.: GreenMind - An architecture and realization for energy smart buildings. In: ICT for Sustainability 2014, (ICT4S-2014) (2014)

35. O'Donnell, J., Corry, E., Hasan, S., Keane, M., Curry, E.: Building performance optimization using cross-domain scenario modeling, linked data, and complex event processing. In: Building and Environment, vol. 62, pp. 102-111 (2013).
36. Palaiokrassas, G., Charlaftis, V., Litke, A., Varvarigou, T.: Recommendation service for big data applications in smart cities. In: Proceedings - 2017 International Conference on High Performance Computing and Simulation, pp. 217-223 (2017)
37. Pauwels, P., Roxin, A.: SimpleBIM: From full ifcOWL graphs to simplified building graphs. In: eWork and eBusiness in Architecture, Engineering and Construction - Proceedings of the 11th European Conference on Product and Process Modelling, pp. 11-18 (2016)
38. Pauwels, P., Zhang, S., Lee, Y. C.: Semantic web technologies in AEC industry: A literature overview. In: Automation in Construction, vol. 73, pp. 145-165 (2017).
39. Pauwels, P.: IFC repository (2015). http://smartlab1.elis.ugent.be:8889/IFC-repo/
40. Puustjärvi, J., Puustjärvi, L.: The Role of Smart Data in Smart Home: Health Monitoring Case. In: Procedia Computer Science, vol. 69, pp. 143-151 (2015)
41. Qu, C., Tao, M., Yuan, R.: A hypergraph-based blockchain model and application in Internet of Things-enabled smart homes. In: Sensors, vol. 18(9) (2018).
42. Rasmussen, M.H., Pauwels, P., Hviid, C.A., Karlshøj, J.: Proposing a Central AEC Ontology That Allows for Domain Specific Extensions. In: 2017 Lean and Computing in Construction Congress, pp. 237-244 (2017)
43. Reinisch, C., Kofler, M., Iglesias, F., & Kastner, W.: Thinkhome energy efficiency in future smart homes. In: EURASIP Journal on Embedded Systems (2011).
44. Rodriguez, M.A., Neubauer, P.: Constructions from dots and lines. In: Bulletin of the American Society for Information Science and Technology, vol. 36(6), pp. 35-41 (2010).
45. SAREF Ontology. http://ontology.tno.nl/saref/
46. Schneider, G.F., Rasmussen, M.D.: TechnicalBuildingSystems/OpenSmartHomeData: First release of Open Smart Home Data Set (Version v1.0.0). Zenodo. (2018). http://doi.org/10.5281/zenodo.1244602
47. SEAS Ontology. https://w3id.org/seas/
48. Spoladore, D., Mahroo, A., Trombetta, A., Sacco, M.: Comfont: A semantic framework for indoor comfort and energy saving in smart homes. In: Electronics, vol. 8(12) (2019).
49. Stavropoulos, T. G., Vrakas, D., Vlachava, D., & Bassiliades, N.: BOnSAI: a smart building ontology for ambient intelligence. In: Proceedings of the 2nd international conference on web intelligence, mining and semantics, pp. 1-12 (2012).
50. Strohbach, M., Saavedra, L.A., Smirnov, P., Legostaieva, S.: Smart home crawler: Towards a framework for semi-automatic IoT sensor integration. In: 2019 Global IoT Summit (GIoTS), pp. 1-6 (2019).
51. Vegter, K.: Cypher Query Optimisations, (2018). https://medium.com/neo4j/cypher-query-optimisations-fe0539ce2e5c.
52. Vicknair, C., Nan, X., Macias, M., Chen, Y., Zhao, Z., Wilkins, D.: A comparison of a graph database and a relational database: A data provenance perspective. In: Proceedings of the 48th Annual Southeast Regional Conference, pp. 1-6 (2010)
53. Workshop Minutes. In: W3C Workshop on Web Standardization for Graph Data, Berlin, Germany (2019). https://www.w3.org/Data/events/data-ws-2019/minutes.html
54. Woznowski, P.R., Tonkin, E.L., Flach, P.A.: Activities of daily living ontology for ubiquitous systems: Development and evaluation. In: Sensors, vol. 18(7) (2018).