Json-Lpg Serialization for Labeled Property Graphs

Ju-Ri Kim, Bong-Hyun Kim

Abstract: ***Labeled Property Graphs (LPG) model applied to NoSQL graph databases has already attracted extensive attention since its remarkable performances on storage and query of graph-structure graphs. In addition, the data interchange based on LPG is increasing in the context of widely applications of Big Data. Hence, a serialization standard for LPG is urgently needed to meet the demand of growing LPG data exchange. This paper proposes a serialization standard called JSON-LPG that is fully compatible with JSON syntax. JSON-LPG provides a novel way to describe data type and shows better performance than conventional serialization formats of graph data. The proposed JSON-LPG not only easily integrates the systems that already use JSON, but also contributes to build interoperable Web applications by LPG data.***

Key words: Data serialization, Data interchange, Labeled Property Graph, GraphML, GraphSON

I. INTRODUCTION

Among numerous NoSQL Database modes, the Graph Databases based on Labeled Property Graph (LPG) have received much attention due to their superior performance on operating the graph-structure data. Nowadays dozens of different graph database products like Neo4j, AllegroGraph, Oracle Spatial and Graph, Graphbase, and Titan, have been widely applied to various Web applications based on graph-structure such as Facebook, Twitter, and so forth [1-4]. The extensive application of graph data models means that the graph data interchange between various heterogeneous systems and platform will be evitable. Therefore, a serialization standard of universal and based on property graph is necessary in order to meet the requirements of data interchange.

However, there is not a succinct and efficient serialization standard of property graph until now. Although there are some optional serialization ways of property graph, such as GraphML and GraphSON, they have some limitations on the expressiveness and efficiency. This paper proposes a concise and efficient serialization scheme called as JSON-LPG, which is a JSON-based and lightweight Labeled Property Graph data format. It is human-readable and have better expressiveness, as well as it is machine-readable and compatible with all syntax of JSON. In addition, the proposed JSON-LPG serialization provides a novel way to declare data type, which not only simplify the representation structure, but also has good expressive power. JSON-LPG overcomes the limitations of conventional serialization ways, which will greatly promote the extensive and deep use on the LPG data.

The rest of this paper is structured as follows. Section 2 introduces the basic structure and characteristics of JSON. The graph model of LPG is explained in this section. Section 3 analyzes the core requirements of JSON-LPG serialization scheme. Comparison with other optional schemes and discussion are provided in section 4. Section 5 summarizes the contributions and puts forth the prospects for further work.

II. BASIC CONCEPTS ABOUT JSON AND LPG

JSON is an open-standard file format of data serialization that uses human-readable text to transmit data objects consisting of key-value pairs, array type data, and any other serializable value. At present, JSON is very common data format used for asynchronous browser-server communications. LPG is an increasingly popular graph data model that has already been supported by numerous graph databases. In addition, the corresponding tools and query language based on LPG have been released over the past years. Especially nowadays LPG has been attracting much attention due to the popularization of NoSQL databases.

2.1 JSON

JSON (JavaScript Object Notation) is a lightweight and text-based data interchange format that can facilitate structured data interchange between all programming languages and different platforms [5]. The text format of JSON is language-agnostic, highly compatible, as well as has similar habits of C language. These features make JSON an ideal describing mode of data exchange. It should be noted that JOSN syntax is not a specification of a complete data interchange, just provides the syntactic framework to which the semantics can be attached [6-7].

A JSON text is a sequence of tokens formed from Unicode code points that conforms to the JSON value grammar, including six structural tokens, strings, numbers, and three literal name tokens. The six structural tokens consist of brace, bracket, colon and comma that is useful in many contexts, profiles, and applications. The three literal name tokens contain true, false and null. The value of a JSON element can be an object, array, number, string, true, false, or null.

Compared with XML

Revised Manuscript Received on May 22, 2019.

JU-RI KIM, Collage of Convergence & Liberal Arts, Wonkwang University, Republic of Korea

BONG-HYUN KIM, Department of Smart IT, U1 University, Republic of Korea

E-mail : cyanic@wku.ac.kr

format, JSON has some significant advantages when developing Web applications [8-9]. JSON file provides a compact way to serialize data, which greatly enhance transmission speed of data. Structural information in JSON file makes JavaScript in Client-side to parse data easily; its support for multiple languages also makes server-side to use data conveniently. These features simplify the code development for both server and client sides, and make program have good maintainability.

2.2 LPG model

LPG is currently popular graph-structure model that mainly used for NoSQL graph databases such as Neo4j. A LPG is a directed, labeled, attributed, and multi-relational graph that consists of vertices, edge, properties, and labels [10-11]. In LPG, vertices usually represent resource or entities, edges represent relationship between resources, and properties are denoted as key-value pairs. LPG have some clear and distinct features, and the following lists its main features:

Ÿ In LPG, each vertex has a uniquely identifiable ID, has arbitrary number of key-value pairs (also known as properties) to characterize it. Vertices may have zero or many labels to describe its type, where the labels are usually used for classification and index.

Ÿ In LPG, each edge is directed edge, and has an outgoing tail vertex and an incoming head vertex. It is similar as vertices that each edge also has a uniquely identifiable ID, has arbitrary number of key-value pairs. The biggest difference from the vertices is that each edge have one and only one label to describe the name of relationship.

The motivation of LPG is to efficient store and fast query against graph data. Hence, LPG graph has some distinct features to enhance such functions. For example, in LPG, both vertices and edges have arbitrary number of key-value pairs, which make the internal structure of graph more compact as well as have better expressiveness. Furthermore, multiple labels of vertices in LPG play a vital important role on both static data representation and dynamic access operation.

III. GOALS AND OBJECTIVES OF SERIALIZATION

As the applications of NoSQL graph database is booming, interchange of graph data based on LPG between various systems and platforms is increasing. Despite several serialization ways of graph data have been proposed in the past years, they are typically restricted in their expressiveness or efficiency when representing LPG data. For example, although GraphML format has powerful expressiveness when describing various graph models, the parse efficiency of GraphML file is slower than JSON file. GraphSON format is verbose owing to the representation of the adjacency list, and especially GraphSON has some limitations when describing the multiple labels of a vertex in LPG.

Recently, the services of Social Networking owning large number of users, such as Facebook and Twitter, are becoming information platforms with huge influence. It means that every day a great deal of graph data need to be exchanged between such services, or stages of a service, and systems specific to areas of applications. This situation means that an efficeint serialization standard for LPG data is necessary. At present, JSON is popular serialization scheme that derived from JavaScript scripting language. JSON is a text-based data interchange format and is often landed as being a lower-overhead to XML format since it just focuses more on content and less on formatting[12-13].

The motivation of this research is to propose a JSON-based data serialization way for LPG model with compact format and better efficiency. In addition, JSON-LPG should satisfy the following design goals:

Ÿ Simplicity: The format provided by JSON-LPG should be well-defined with a succinct and clear structure. It should be not only human-readable, but also machine-readable. It should be easy to parse and use by both server and client sides.

Ÿ Efficiency: The definition of JSON-LPG should take full advantage of characteristics of JSON syntax to build compact file format, which should make the parse and interpretation of serialization file more efficient, and reduce the time-consuming during the transmission and parse.

Ÿ Compatibility: A JSON-LPG document is always a valid JSON document. JSON-LPG should be completely compatible with JSON syntax, which will ensure all of the standard JSON libraries and tools work seamlessly with JSON-LPG documents.

Ÿ Expressiveness: All the features of LPG should be accurately described through JSON-LPG without any ambiguities. JSON-LPG must ensure that all the real world LPG data can be expressed effectively.

IV. JSON-LPG SERIALIZATION

4.1 General terminology

The general terminology of JSON-LPG consists of basic terminology and exclusive terminology. The basic terminology is defined as JSON terminology since JSON-LPG is completely compatible with JSON syntax. The exclusive terminology as the extension of JSON terminology is defined in order to enhance the representation power of JSON-LPG, used to describe data types, graph mode, name space, and so forth[14-16].

4.1.1 Basic JSON terminology

JSON-LPG is compatible to all JSON syntax, including all the terminology terms defined in JSON. The following explains the basic terminology terms:

JSON object: An object structure is typically denoted as a pair of curly brackets surrounding zero or more key-value pair. A single colon is used for separating the key from its value, and a single comma is used for separating two key-value pairs.

Array: An array structure is denoted as square brackets surrounding zero or more values. Values are separated by commas.

String: A string is a sequence of zero or more

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Unicode characters, wrapped in double quotes. The backslash escapes are valid.

Number: A number, an arithmetical value, is similar to that used in most programming language. However, the octal and hexadecimal format are not supported in JSON-LPG.

True and false: Two constant values for Boolean variables.

Null: The null value is usually used to represent an undefined or forgotten value.

4.1.2 Exclusive JSON-LPG terminology

In order to describe typical features of LPG accurately, JSON-LPG provides some exclusive teminology terms used for description the characteristics of graph, vertices and edges in LPG.

The terminology terms used for describing graph include “mode”, “declarationOftype”. The term “mode” is used for describing graph mode just like directed, or, undirected graph. The term “declarationOftype”is a Boolean type, and it is used to show whether describe selected data types explicitly.

The exclusive terminology terms of vertices description include "identifier", "labels" and "properties". The term "identifier" is used to identify each node in the LPG. The term "labels" is a JSON array that is used for describing multiple labels of a vertex. For a vertex without labels, the key-value pair about “labels” will be left out. The term "properties" is an object structure that contains all the property information of a vertex.

The exclusive terminology terms of edges description include "identifier", "relation", "source", "target" and "properties". The term "relation" is used to describe the name of edge, where each edge has one and only one value of “relation”. The term "source" shows the outcoming node of this edge, which value is the "identifier" of node. The term "target" shows the incoming node of this edge, which value is also the "identifier" of node. The two terms "identifier" and "properties" have similar meaning to corresponding terms of vertices.

4.2 The declaration of data type

JSON-LPG does not provides any exclusive terminology terms to declare data types, only provides two optional ways to describe data type. One way is to describe data type explictly; the other way is to omit the description of data type. The value of exclusive terminology term “declarationOftype” determines whether to describe data type or not. If its value is false, it means the serialization file does not contain the description about data type. Otherwise, the serialization scheme will specify data type for each property value of vertices and edges. It is important to note that only the property value of vertices and edges have data type description, exclusive terminology terms not.

JSON-LPG provides a novel way to describe data type, in the form “key @type: value”. Namely, the description of data type is contained in “key” part of “key: value” pair, where the character “@” is used to divide the key name and data type. JSON-LPG only provides a format to describe data type, whereas the used keywords of data type entirely depend on the users’ choice. The conventional description method of data type is to add additional “key: value” pair to describe it as shown in figure 1(a). In this paper, the novel description method of data type, as shown in figure 1(b), not only simplify the structure, but also is compatible with JSON syntax. Besides, it enhances the expressiveness while meeting the requirement of compatibility.

```
"name": {
    "value": "Tom Brown",
    "type": "string"
},
"age": {
    "value": 18,
    "type": "integer"
}
```

(a) Conventional method

```
"name @string": "Tom Brown",
"age @integer": 18
```

(b) A novel method

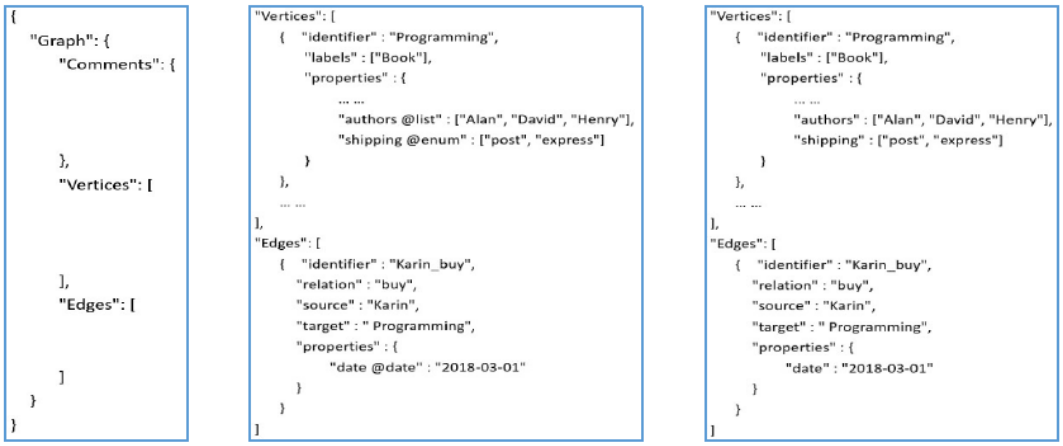
```
"name": "Tom Brown",
"age": 18
```

(c) No description of data type

Figure 1 The description methods of data type

4.3 Basic structure

A Label Property Graph is a JSON object that consists of three properties modules: comments, vertices and edges modules. The properties of whole graph should be described in comments module that is a JSON object. The properties of vertices and edges are declared in vertices and edges modules that are both JSON array structures. The whole structure of JSON-LPG is described as Figure 2(a).



(a) Basic structure (b) declarationOftype is true (c) declarationOftype is false

Figure 2 Structure of JSON-LPG

JSON-LPG determines whether to describe data type according to the value of term "declarationOftype". If its value is true, that means data types of values are described explicitly in the form "key @ type: value", otherwise in the traditional JSON form "key: value" where data types are omitted. Figure 2(b) and Figure 2(c) show the difference. For example, in Figure 2(b), it shows the information of a book, including author and shipping methods. The type of author is "list" that means the array is a group of ordered values. The type of shipping methods is "enum" that means the array is a group of enumeration values.

It must be noted that the exclusive terminology terms do not need to describe their value's type, and they are string type by default such as "identifier", "labels", "relation", "source" and "target". Only in the properties description of vertices and edges, data types of properties value can be declared explicitly.

V. DISCUSSION AND ANALYSIS

This paper proposes a lightweight LPG serialization schema called as JSON-LPG, which is fully compatible with JSON syntax. JSON-LPG has a clear and succinct structure, which makes the serialization formats to be read and written easily for both humans and machines. In addition, JSON-LPG can benefit from various libraries and tools based on JSON due to good compatibility.

Especially, JSON-LPG serialization method provides a novel way to describe data type in the form "key @type: value". Compared with conventional declaration way of data type, the new method not only describes the data type, but also reduces the file size. In addition, the way to describe data type does not destroy the compatibility with JSON syntax, at the same time, enhancing the expressive power.

JSON-LPG has significant advantages in various aspects contrast to classical graph serialization methods such as GraphML and GraphSON. The following discusses performance comparison between GraphML, GraphSON and JSON-LPG.

Ÿ Human-readability

JSON-LPG serialization provides a human-readable format to describe LPG data. By comparison, GraphSON format is very verbose owing to the application of adjacent list, which adds the difficulty of human read. Generally speaking, both GraphML and JSON-LPG formats show a little better performance on human-readability than GraphSON due to the conciseness of data description.

Ÿ Machine-readability

Compared the three serialization formats for Labeled Property Graph, JSON-LPG has obvious advantages on machine readability. Since JSON-LPG provides compact format to describe data by using clear structure, the structure of data block make machine parse more easily. In other words, JSON-LPD focuses more on data description rather than format representation.

Ÿ File size

JSON-LPG serialization hold the smallest file size than the serialization ways of GraphML and GraphSON for LPG data. Since GraphML applies many makeup language to define file format and GraphSON stores a large number of redundant graph structure information, these both make their serialization files own bigger size.

Ÿ Expressive power

JSON-LPG can express all the real LPG data accurately and efficiently. GraphML can express LPG data and has stronger expressive power when describing diversity of graphs, such as mixed multigraphs and hypergraphs. However, GraphSON has some limitations when describing multiple labels of a node in LPG.

Ÿ Parse difficulty

Since JSON-LPG is completely based on JSON structure and fully compatible with JSON syntax, this makes JSON-LPG more easily to parse and use, as well as reduces the program development difficulty. Besides, the simple format of JSON-LPG file makes it has faster parse speed than GraphML and GraphSON.

Taken together, since JSON provides an ideal foundation for JSON-LPG, it makes JSON-LPG has the same advantages as JSON, such as good machine readability, smaller file size and faster parse speed. In addition,

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

JSON-LPG can benefit from a large number of available tools for reading, parsing and processing JSON-based data.

VI. CONCLUSION

LPG, one of current popular graph structure models, has abstracted much attention due to its excellent performance in NoSQL graph databases. Especially in the context that the applications of Social Networks and Linked Open data based on graph models is springing up, the requirement of LPG data interchange is increasing prominent. A serialization scheme for LPG is urgent demand to meet the requirement of data exchange.

This paper proposes a JSON-based lightweight serialization way for LPG called JSON-LPG. This serialization way provides a novel format to describe data type by using the form "key @type: value", which not only compacts the data format as well as enhances expressiveness, but also completely adheres to JSON syntax without the introduction of new syntax structure. For LPG, the proposed JSON-LPG format has succinct structure, fine expressive power and better machine readability as opposed to other graph serialization formats, which will contribute to the deeper and wider applications of LPG data.

ACKNOWLEDGMENT

This paper was supported by Wonkwang University in 2018.

REFERENCES

1. Angles R, Gutierrez C. Survey of graph database models [J]. ACM Computing Surveys (CSUR), 2008, 40(1): 1.
2. Han J, Haihong E, Le G, et al. Survey on NoSQL database [C], Pervasive computing and applications (ICPCA), 2011 6th international conference on. IEEE, 2011: 363-366.
3. Strauch C, Sites U L S, Kriha W. NoSQL databases [J]. Lecture Notes, Stuttgart Media University, 2011, 20.
4. Das S, Srinivasan J, Perry M, et al. A Tale of Two Graphs: Property Graphs as RDF in Oracle [C], EDBT. 2014: 762-773.
5. Bray T. The javascript object notation (json) data interchange format [J]. 2017.
6. Peng D, Cao L, Xu W. Using JSON for data exchanging in web service applications [J]. Journal of Computational Information Systems, 2011, 7(16): 5883-5890.
7. Pezoa F, Reutter J L, Suarez F, et al. Foundations of JSON schema [C], Proceedings of the 25th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2016: 263-273.
8. Nurseitov N, Paulson M, Reynolds R, et al. Comparison of JSON and XML data interchange formats: a case study [J]. Caine, 2009, 9: 157-162.
9. Maeda K. Performance evaluation of object serialization libraries in XML, JSON and binary formats [C], Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on. IEEE, 2012: 177-182.
10. Rodriguez M A, Neubauer P. Constructions from dots and lines [J]. Bulletin of the Association for Information Science and Technology, 2010, 36(6): 35-41.
11. Robinson I, Webber J, Eifrem E. Graph databases: new opportunities for connected data[M]. " O'Reilly Media, Inc.", 2015.
12. Bray T. The javascript object notation (json) data interchange format [J]. 2017.
13. Peng D, Cao L, Xu W. Using JSON for data exchanging in web service applications [J]. Journal of Computational Information Systems, 2011, 7(16): 5883-5890.
14. Pezoa F, Reutter J L, Suarez F, et al. Foundations of JSON schema [C], Proceedings of the 25th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2016: 263-273.
15. Nurseitov N, Paulson M, Reynolds R, et al. Comparison of JSON and XML data interchange formats: a case study [J]. Caine, 2009, 9: 157-162.
16. Maeda K. Performance evaluation of object serialization libraries in XML, JSON and binary formats [C], Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on. IEEE, 2012: 177-182.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication