GridKG: Knowledge Graph Representation of Distribution Grid Data

Yashar Kor
Electrical and Computer Engineering
University of Alberta
Email: yashar@ualberta.ca

Liang Tan
Electrical and Computer Engineering
University of Alberta
Email: tan3@ualberta.ca

Marek Z. Reformat
Electrical and Computer Engineering
University of Alberta
Email: reformat@ualberta.ca

Petr Musilek
Electrical and Computer Engineering
University of Alberta
Email: pmusilek@ualberta.ca

***Abstract*—Distribution grid systems are complex networks containing multiple pieces of equipment. All of them interconnected, and all of them described a variety of pieces of information. A knowledge graph provides an interesting data format that allows us to represent information in a form of graphs, i.e., nodes and edges – relations between them. In this paper, we describe an application of a knowledge graph to represent information about a power grid. We show the main components of such a graph – called GridKG, a simple process of identifying electrical paths, and a few examples of grid analysis related to primary switches.**

***Index Terms*—distribution grid, knowledge graph, electrical path, grid analysis**

I. Introduction

A distribution level power system is a complex and connection intense structure. It can be characterized by the presence of multiple components of different type – many of them ‘small’ and of a low economical value when compared with elements of a transmission level system – that are connected in a chain-like mode. Such a nature of a distribution grid makes it quite tedious to oversee, analyze, and maintain.

A power utility keeps information about its grid in relational databases spread access multiple organizational units. In order to ‘see’ a part of the system, i.e., components, their types and parameters, and details of their physical connections, as well as information about linked customers and their locations, a number of operations related to accessing few databases and performing multiple operations on tables containing information about components of interest need to be performed.

Recently, a new type of databases that offer a different data representation format become more and more popular. They are graph-based databases that represent information as network of nodes and relations between them. Such databases provide a number of benefits:

- easiness of data integration from multiple data sources;
- easiness of expansion, i.e., ability to continuously append new pieces of data;
- simplicity of accessing individual pieces of information and performing queries, especially in the context of connections/relations between data nodes;
- openness to develop and execute variety of algorithms on graphs in order to gain additional information and enhancing graphs, for example, computing a length of sequence of specific types of nodes.

This paper shows how the power grid’s components can be stored in a format of the knowledge graph in a graph database, Neo4j. We provide details related to concepts and relations between them that constitute nodes and edges of a graph. We briefly describe a graph – called GridKG – representing a fragment of the distribution grid. A few examples showing how such a database can be utilized are described.

The paper’s main contribution is constructing a knowledge graph that integrates information about the topology of the power system with meta data about equipment and customers. We enhanced the knowledge graph with the data generated by algorithms we developed to identify upstream and downstream devices, as well as the number of customers connected to them. This additional data leads to a holistic view of the system. All this would allow us to gain further insight into the system’s characteristics while analyzing the grid.

II. Background and Related work

Before we introduce our knowledge graph, GridKG, we provide a short description of the concept of knowledge graph, and a few examples of utilization of graph-based representation of data in power systems.

A. Knowledge Graphs

A knowledge graph is basically a data management system which combines various types of data and utilizes graphs to represent information and knowledge [1]. Initially introduced by Google, the knowledge graph concept was utilized to optimize search engine performance with the information gathered from various sources. Knowledge graphs such as BabelNet [2], DBpedia [3], WordNet [4], Microsoft’s Probase [5] and Google Vault [6] are created focusing on text-based extraction of data from the web content. The existing generic knowledge graphs prove their usefulness in applications such as semantic search, and information fusion from various sources.

The Resource Description Framework (RDF) data format [7] introduced by the Semantic Web as a standard for Linked Open Data is a popular graph-based data format in which each piece of data is stored as a RDF triple that contains: two entities, two nodes in a graph, called a subject and an

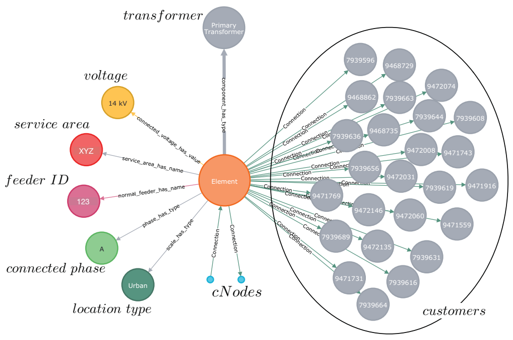

Figure 1. RDF triples with *Element* as their subject

object; and a relation between them, an edge in a graph, called a property. Processing data represented as knowledge graphs in an RDF format is generating a lot of attention. There are multiple works focusing on different aspects related to RDF- based Knowledge Graphs: from their construction [8], via storage [9], querying strategies [10], [11] and extracting information [12], to applications [13], [14], just to mention a few. The fact that subjects of triples could be also objects of another triples, and vice versa, means that we deal with a network of entities highly interconnected via properties.

A single RDF-triple <subject-property-object> can be perceived as a feature of an entity identified by the subject. In other words, each single triple is treated as a feature of its subject. Multiple triples with the same subject constitute a description of a given entity. A set of few RDF triples with *Element* as their subject is presented in Fig. 1. As it can be seen, each triple provides a piece of description about *Element*: its type – *primary transformer*, its connected voltage: *14kV*, a service area of its location: *XYZ*, a feeder to which it is attached: *123*, a connected phase: *A*; and all customers connected to it.

Quite often a subject and object of one triple can be involved in multiple other triples, i.e., they can be objects or subjects of other triples. In such a case, multiple definitions can share features, or some of the features can be centres of other entity descriptions. All interconnected triples constitute a network of interleaving definitions of entities.

B. Related Work

In the realm of power industry, few studies have focused on domain-specific knowledge graphs application in power grid operation. Industrial application of knowledge graph at Siemens was an important step toward intelligent engineering and manufacturing which could highly improve workflow efficiency and data accessibility [15]. Enterprise-level information integration framework, enhanced power equipment management and improved efficiency in querying and classifying relevant information and updating product contents in real-time [1].

In [16], single- and multi-threading methods for analysis of grid energization are designed. It was demonstrated that the graph database had better efficiency compared to a relational database for the power grid analysis. Some recent works investigated various relation extractions between specific power equipment entities. In [17] automatic framework is developed to extract ontological information which facilitates the sharing of multi-source heterogeneous power grid equipment data.

The data island problem for smart grid equipment was solved by considering inter equipment relationship and grid equipment multifaceted nature [18]. In [19] authors leveraged evolving graph databases to enhance the process of adding new equipment and potential expansion of database. Multi-source information fusion method was developed to reduce information redundancy and improve accuracy of fusion [20].

The relationship between the grid topology and power system reliability using the graph theory and statistical analysis was investigated by [21] while in [22] electric devices abnormality detection using knowledge graph was the focus of study. All these efforts and studies confirm the increasing interests and needs for knowledge graph in the power grid domain.

III. GridKG Vocabulary: Concepts and Relations

One of the most critical activities required for constructing any knowledge graph is 'building' so called vocabulary. It is a set of concepts and relations that are used to name nodes and edges of a graph. Due to a space limitation, we present the most important categories of concepts (for nodes) and relations (for edges).

A. Concepts

Classes of concepts should reflect items and elements that constitutes main pieces of data that a graph suppose to represent. In the case of a distribution grid, it has been decided to use a concept of *Element* as a basic component representing any asset of the system. An *Element* has four attributes: *id* that is the same as the asset ID assigned to it by the utility, *no_of_customers* that are connected to a downstream path, and two coordinates *x* and *y* as identifiers of its geographical location.

Additionally, there are a number of concepts that provide information about *Elements*. Any piece of information about an *Element* is provided as a node connected/linked to it. They are:

cNode – a fictitious connection point, *Element* is connected to two of them: one upstream, and one downstream ('decided' after running Algorithm 1, see below); there are two unique nodes – *ENode* representing the last/end node, and an *SNode* that is the staring node;
Component Type – a type of *Element*, for example, *Primary Transformer*, *Primary fuse*, *Capacitor*, *Elbow*;
Feeder – an identification of a feeder to which *Element* is connected;
Service Area – a name of a service area where *Element* is located;
Connected Voltage – a voltage value of connected *Element*;
Phase Type – a phase to which *Element* is connected;
Location – a type of location of *Element*;
Customer – a no of customers (downstream) connected to *Element*;

A few illustrations how nodes of these categories are interconnected are provided below.

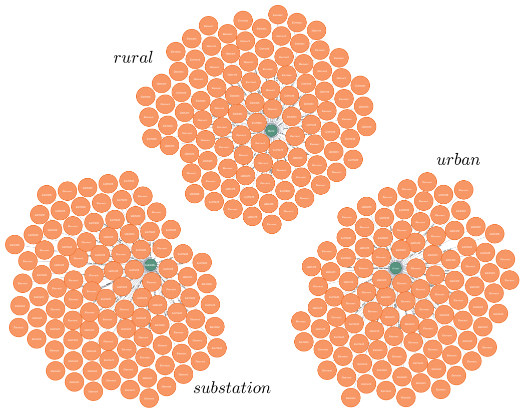

Figure 2. *Elements* by location type

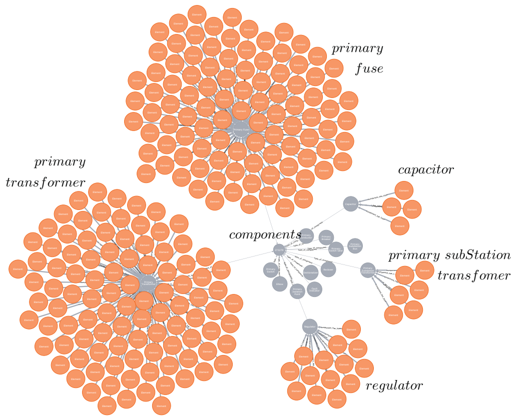

Figure 3. *Elements* by component type

B. Relations

Relations define a type of connection that exists between nodes. We are inclined to say that these connection constitute an essence of a graph-based representation. They allows to show how elements (nodes) are connected, and how they are linked to categories representing additional pieces of information. Here we have:

type of component – links *Element* to *Component Type*;
name of feeder – links *Element* to *Feeder*;
name of service area – links *Element* to *Service Area*;
connected voltage has value – links *Element* to *Connected Voltage*;
value of no of phases – links *Element* to *No of Phases*;
phase has type – links *Element* to *Phase Type*;;
connection – links *Element* to *cNode*;

Let us take a look at a few examples how elements are represented in the GridKG. The first example is already illustrated in Fig. 1. Another one is shown in Fig. 2. It displays *Elements* sorted by three different locations. An illustration of different types of *Elements* is included in Fig. 3.

The figures exhibit a characteristic feature of GridKG – nodes that represent electrical components and pieces of information describing them, as well as connections that link them together. Further, the connections – edges in a graph – clearly define relations that exist between nodes, i.e., components themselves, and components and information.

IV. GridKG and Grid Analysis

The availability of having a graph-based representation of a grid allows us to easily preform a number operations on the data. Many of these operations take advantage of the easiness to see the data as sequences of connected elements.

In order to illustrate one possible such operation, we focus on a process of identifying electrical paths in the system. Following that, we show a few samples of utilization of paths created in GridKG.

A. Electrical Paths

An electrical path in a graph-based representation of grid is perceived as a sequence of tuples composed of two triples: $cNode_x$–*Connection*→$Element_p$ and $Element_p$–*Connection*→$cNode_y$. Therefore, a process of identifying paths means 'stitching' together multiple tuples in a way that $cNode_y$ of the predecessor tuple matches $cNode_x$ of the successor tuple.

The algorithm for identifying paths is shown below (Algorithm 1). At the beginning, all element-breakers and *cNodes* connected to them are put into a queue **Q** of elements to inspect for the purpose of finding other connected elements. Then a Breadth-First search is applied to find all adjacent elements and *cNodes* which satisfy the condition that the elements are not opened switches. A variable *level* is used as a property of elements to 'keep' track of elements' positions in the paths. For each element, the algorithm modifies the direction of edges from element of lower value of *level* to elements with higher value of *level*. The algorithm terminates when the queue **Q** is empty.

One of the additional benefits of identifying paths and their direction is generating new information about elements and adding it to the existing element nodes. For instance, by implementing a simple algorithm, we compute a number of customers for each element by a bottom-up approach. The number of customers for each element is determined by aggregating all the downstream customers. This highly can be beneficial to estimate the number of customers out of power after isolation of a grid by a specific protection device.

B. Primary Switches on Path

Once electrical paths are identified, we can use GridKG to learn more about interconnection between its components. One of possible ways of learning more about the grid is finding out all downstream (and upstream) elements and connections from a given element of the system.

An example of a query in Neo4j query language Cypher is shown below. We provide a starting point/element and the query returns all downstream connected elements, Fig. 4. Additionally, it gives us a length of the electrical path – in our case it is equal to 911.34m. In Fig. 4, we see information

Algorithm 1 Multi-Source Path Search

```
1: Initialization: Queue $\mathbf{Q}$ = [], $level$ = 0, Sets: $\mathbf{S}$ ={}, nextElements = {}
2: $\mathbf{Q} \leftarrow$ list of pairs (cNode,breakers) i.e. $\mathbf{Q} \leftarrow \{ (n_1, b_1), (n_2, b_2), \ldots ,(n_n, b_n) \}$

3: while $\mathbf{Q}$ is not empty do
4:     $size \leftarrow \mathbf{Q}.size()$
5:     for from 1 to $size$ do
6:         $(cNode, element)$ = $\mathbf{Q}.poll()$
7:         if $element.id \in \mathbf{S}$ then                    ▷ Element visited
8:             continue
9:         $\mathbf{S}.add(element.id)$
10:        $element.level \leftarrow level$
11:        $next\_cN \leftarrow$ GetNextNode($element$, $cNode$)
12:        nextElements $\leftarrow$ GetNextEl($element$, $next\_cN$)
13:        for $next\_E$ in nextElements do
14:            SetConDirect($element$, $next\_cN$, $next\_E$)
15:            if not ($next\_E$ = Switch & $next\_E.state$ = Opened) then
16:                $\mathbf{Q}.add(next\_cN, next\_E)$
17:            end_if
18:        end_for
19:    end_for
20:    $level \leftarrow level + 1$
21: end_while
22: return
```

about types of elements, as well as their connected voltage and phase details (for clarity, some details are not shown).

```
MATCH
    path = (source:Element{id:1235})
    -[:Connection*]->(leaf:Element)
    -[c:Connection]->(eNode:ENode)
RETURN
    path ,
    reduce(total = 0, e IN nodes(path) |
        CASE
        WHEN e.length IS NOT NULL
        THEN total + e.length
        ELSE total
        END
    ) AS totalLength
```

C. Primary Switches and Conditions

Another type of analysis could lead to learning more about electrical path in the context of a number of protective devices present in the path. Such an information is very easy to retrieve from GridKG.

Below, we include a query that provides – as a result – a number of primary switches on the specified path, as well as all components of this path, Fig. 5. We can identify the first and all subsequent protective devices on the upstream path from a specified element (top-left corner of the figure) to the primary breaker (down-right corner).

Besides localization of the switches, we obtain more information about the path itself – its components, and information about them.

```
MATCH
    path = (source:Element{id:1456})
    <-[:Connection*]-(root:Element)
    <-[c:Connection]-(sNode:SNode)
RETURN
    path ,
    [ element IN nodes(path)
        WHERE
         (element)-[:component_has_type]
         ->(:Component_type
         {component_type: 'Primary Switch'})
    ] AS protectiveElements
```

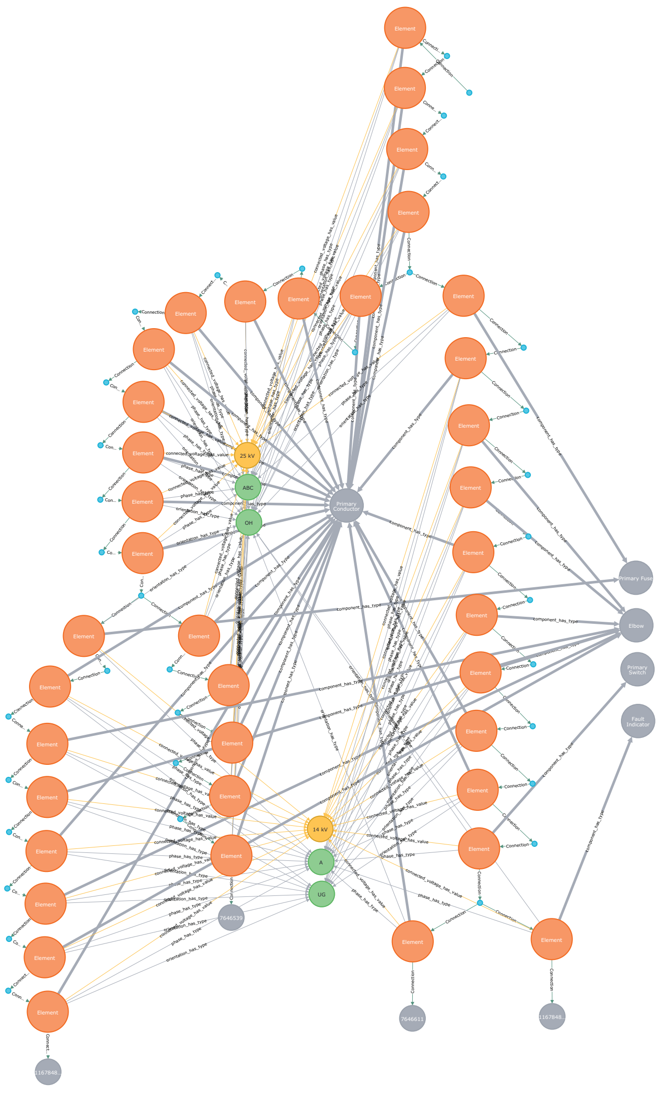

Figure 4. *Elements* of downstream paths: their types identified by links to gray circles where each circle represents a different type of electrical component; their voltages – yellow circles; and their phases and orientation – green circles.

D. Elements and Conditions

As the last example of utilization of GridKG, we show a bit more involving query. This time, we want to know about the existence of primary switches that are in the 'XYZ' service area, connected to 14 kV, and located on paths that deliver power to more than 100 customers.

The query is below, and the obtained data is illustrated in Fig. 6.

```
MATCH
    (e:Element)
    -[:component_has_type]
    ->(c:Component_type
    {component_type: "Primary Switch"})
```

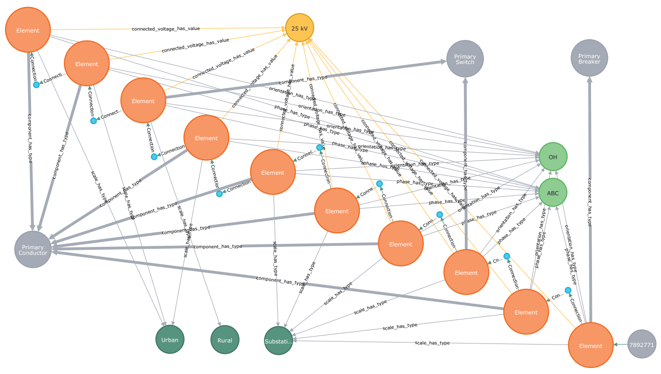

Figure 5. Location of Primary Switch/protective devices on the upstream path from top-left element to the primary breaker (down-right corner).

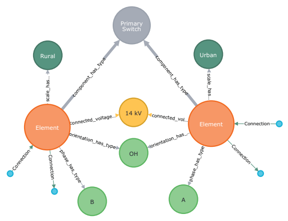

Figure 6. Primary switches in the 'XYZ' service area that connected to 14 kV and provide power to more than 100 customers downstream: they are of different scale, connected to different phases.

```
WHERE
      (e)-[:service_area_has_name]
      ->(:Service_area_name
      {service_area_name: "XYZ"})
AND
      (e)-[:connected_voltage_has_value]
      ->(:Connected_voltage_value
      {connected_voltage_value: "14 kV"})
AND  e.num_customer > 100
RETURN  e
```

V. Conclusion

Today's distribution grids are complex networks constituted of multiple components. Power utilities collect and store, in relational databases, large amount of information about the grids' elements from transformers to individual poles. It is important for them to be able to have quickly access the data describing components, as well as connections and relations between them.

We propose to use knowledge graphs as a suitable format for representing grid data. We describe some of the categories of nodes designed for representing different electrical elements and conceptual information describing those elements. We also define a number of relations between elements/concepts that are linked to edges connecting nodes of the graph.

Finally, we illustrate utilization of a distribution grid knowledge graph. We propose an algorithm for identifying electrical paths in the grid. Further, we include a few graph queries that take advantage of the identified paths and allow for: determining a length of downstream path from a specific element; determining a sequence of switches/protective devices on a given upstream path; as well as a set of switches that satisfy a condition related to downstream components.

References

[1] Y. Tang, T. Liu, G. Liu, J. Li, R. Dai, and C. Yuan, "Enhancement of power equipment management using knowledge graph," in *2019 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)*, pp. 905–910, IEEE, 2019.

[2] R. Navigli and S. P. Ponzetto, "Babelnet: Building a very large multilingual semantic network," in *Proc. of the 48th annual meeting of the association for Comput. linguistics*, pp. 216–225, 2010.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*, pp. 722–735, Springer, 2007.

[4] "A lexical database for english." https://wordnet.princeton.edu/. Accessed: 2020-08-11.

[5] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *Proc. of the 2012 ACM SIGMOD Int. Conf. on Management of Data*, pp. 481–492, 2012.

[6] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *Proc. of the 20th ACM SIGKDD Int. Conf. on Knowl. discovery and data mining*, pp. 601–610, 2014.

[7] "Resource Description Framework." https://www.w3.org/RDF/. Accessed: 2018-10-25.

[8] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, vol. 8, no. 3, pp. 489–508, 2017.

[9] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. on Knowl. and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[10] H. Arnaout and S. Elbassuoni, "Effective searching of rdf knowledge graphs," *J. of Web Semantics*, vol. 48, pp. 66–84, 2018.

[11] R. Verborgh, M. Vander Sande, O. Hartig, J. Van Herwegen, L. De Vocht, B. De Meester, G. Haesendonck, and P. Colpaert, "Triple pattern fragments: a low-cost knowledge graph interface for the web," *J. of Web Semantics*, vol. 37, pp. 184–206, 2016.

[12] U. Lösch, S. Bloehdorn, and A. Rettinger, "Graph kernels for rdf data," in *Extended Semantic Web Conf.*, pp. 134–148, Springer, 2012.

[13] T. Ruan, L. Xue, H. Wang, F. Hu, L. Zhao, and J. Ding, "Building and exploring an enterprise knowledge graph for investment analysis," in *Int. Semantic Web Conf.*, pp. 418–436, Springer, 2016.

[14] P. Szekely, C. A. Knoblock, J. Slepicka, A. Philpot, A. Singh, C. Yin, D. Kapoor, P. Natarajan, D. Marcu, K. Knight, *et al.*, "Building and using a knowledge graph to combat human trafficking," in *Int. Semantic Web Conf.*, pp. 205–221, Springer, 2015.

[15] T. Hubauer, S. Lamparter, P. Haase, and D. M. Herzig, "Use cases of the industrial knowledge graph at siemens.," in *Int. Semantic Web Conf. (P&D/Industry/BlueSky)*, 2018.

[16] B. Kan, W. Zhu, G. Liu, X. Chen, D. Shi, and W. Yu, "Topology modeling and analysis of a power grid network using a graph database," *Int. J. of Comput. Intell. Systems*, vol. 10, no. 1, pp. 1355–1363, 2017.

[17] Z. Su, M. Hao, Q. Zhang, B. Chai, and T. Zhao, "Automatic knowledge graph construction based on relational data of power terminal equipment," in *2020 5th Int. Conf. on Comput. and Communication Systems (ICCCS)*, pp. 761–765, IEEE, 2020.

[18] H. Huang, Y. Chen, B. Lou, Z. Hongzhou, J. Wu, and K. Yan, "Constructing knowledge graph from big data of smart grids," in *2019 10th Int. Conf. on Information Technology in Medicine and Education (ITME)*, pp. 637–641, IEEE, 2019.

[19] A. Perçuku, D. Minkovska, and L. Stoyanova, "Modeling and processing big data of power transmission grid substation using neo4j," *Procedia Comput. science*, vol. 113, pp. 9–16, 2017.

[20] Y. Yang, Z. Chen, J. Yan, Z. Xiong, J. Zhang, Y. Tu, and H. Yuan, "Multi-source heterogeneous information fusion of power assets based on knowledge graph," in *2019 IEEE Int. Conf. on Service Operations and Logistics, and Informatics (SOLI)*, pp. 213–218, IEEE, 2019.

[21] Y. Xu, T. Yu, and B. Yang, "Reliability assessment of distribution networks through graph theory, topology similarity and statistical analysis," *IET Gen., Transm. & Dis.*, vol. 13, no. 1, pp. 37–45, 2018.

[22] B. Cui, "Electric device abnormal detection based on iot and knowledge graph," in *2019 IEEE Int. Conf. on Energy Internet (ICEI)*, pp. 217–220, IEEE, 2019.