Heterogeneous Social Linked Data Integration and Sharing for Public Transportation

Wei Zhao, Bing Zhou, and ChaoYang Zhang

1 School of Information Engineering, Zhengzhou University, Zhengzhou 450000, China
2 Cooperative Innovation Center of Internet Healthcare, Zhengzhou University, Zhengzhou 450000, China
3 School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou 450002, China

Correspondence should be addressed to Wei Zhao; iezhaowei@zzu.edu.cn

Received 25 January 2022; Accepted 12 April 2022; Published 11 May 2022

1. Introduction

Data sharing for public transportation could address the complex city challenges and implement the efficient and effective management of city spaces, and it could be an effective way to reduce pollution, carbon emissions, global climate change, and even manage health risks in smart cities [1]. Data sharing can involve data such as bus routes, live bus arrival times, train routes, car parking availabilities, and live traffic routes. In addition, a range of data acquisition methods exists. Furthermore, there exist different system sources and data storage formats. This makes the data composition unstructured, which causes heterogeneity in the data composition. Furthermore, heterogeneous data make it difficult to share public transportation data [2–6].

To address these challenges, many studies have been made over the past years. Some researchers tried to integrate heterogeneous data by Web API [7, 8]. They built a virtual integration way in which a unified query interface is provided to a large number of heterogeneous data sources, but this method is limited to popular programming languages and cannot get over the challenges of evolving API that need to modify the existing client codebase [9].

To realize the unified management of heterogeneous data in data applications, one method is to use ETL tools to process heterogeneous data in a unified form. However, this method is costly, and changes in data and analysis requirements will cause the original ETL process to fail [10].

Other studies have shown that the semantic web is "a common framework that allows data to be shared and reused". The semantic web is based on the Resource Description Framework (RDF) and the Web Ontology Language (OWL). It uses a set of metadata to describe data, allowing machines to understand and interpret the data. This makes it easier to integrate and share data, which is particularly useful for public transportation data.
across application, enterprise, and community boundaries [11].” Semantic web technology [12] promotes data integration from multiple heterogeneous data sources, enables the development of information-filtering systems, and supports knowledge discovery tasks. It helps to integrate diverse data and share these data through the semantic web. However, most of the content on the World Wide Web has not yet been marked up to comply with the semantic web specification. Therefore, how to automatically add tags that conform to the semantic web specification to the existing web content is one of the difficult problems facing the practical application of the semantic web.

To provide personalized sharing of public transportation data between different users through integrating and sharing these heterogeneous data, a new approach is proposed in this study. The proposed approach mainly has three key elements: a data processing method, a CSV to RDF (Resource Description Framework) Mapping File algorithm, and a system framework. Firstly, in order to integrate heterogeneous data, a data processing method is proposed. This data processing method includes the process of collecting, organizing, and normalizing data. Data processing is the basic element of data and data system management and involves the management process of the full life cycle of data. Based on the above point, the main focus is on unstructured data integration. C2RMF (CSV to RDF Mapping File) algorithm is designed, which mainly converts CSV files into RDF files and converts them into a unified data format for subsequent processing. Third, to share these integrated heterogeneous data, a system framework is proposed. This framework mainly performs unified data storage, query, and visualization of the unified transformed files. The main contributions of this study are summarized as follows:

(i) The proposed data processing method involves collecting, integrating, generating, and sharing the heterogeneous public transportation data.

(ii) Based on the proposed method, the newly developed C2RMF algorithm for integrating heterogeneous data is proposed.

(iii) A novel system framework for integrating and sharing these heterogeneous public transportation data is proposed.

The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 describes the proposed methodology in detail. Section 4 describes the system framework. Section 5 describes the system implementation process. The next section describes the experimental environment, data, process and results, and analysis. The conclusions and recommendations are given in the final section [13].

2. Related Work

2.1. RDFS and OWL. Integration and sharing of heterogeneous public transportation data is an important problem in recent years. Previous research and related technologies had solved users’ needs to a certain extent, but there are still some problems [7–12]. Recently, RDF (Resource Description Framework) has been widely used to expose, share, and connect pieces of data on the web. RDF is a W3C recommendation graph database model. RDF provides a model of nodes and relationships and is a more general model for the World Wide Web. RDF includes a range of statements, with each statement describing a resource. Every statement includes three parts: a subject, a predicate, and an object. The subject represents the resource; the predicate mainly represents a relationship between the subject and object and the property of that resource; the object represents the value of that property.

Although RDF provides the ability to create a graph of data, it is the RDF schema (RDFS) that supplies the building blocks for more complex vocabularies by enabling the definition of data types and structures within an RDF graph [14]. RDFS provides “all that is needed for interoperability of the vast amount of data on the web [15].” The RDFS can define a set of word sets that can be clearly described based on RDF resources and can be used to describe the subclass or superclass relation, subattribute or superattribute, domain and range of attributes, and instance constraints of the class. Another RDF data model named OWL (Web Ontology Language), which is a W3C recommendation, is a family of knowledge representation languages for authoring ontologies. It can describe more complex logical relationships between concepts. OWL extends RDFS and provides a more descriptive schema layer that can be used where the basic definitions afforded by RDFS are not expressive enough [16].

With the help of the data interconnection network constructed by RDFS and Web Ontology Language (OWL), they are combined with different heterogeneous datasets published on the web. However, they often lack a unified data operation mode for information storage and retrieval of unstructured data.

2.2. Social Linked Data. Solid is also proposed to provide important capabilities for web-based data-sharing systems for public transportation as Solid is based on semantic web and RDF technologies, and it provides a unified data operation mode. In Solid, every user can store their dataset in an online storage space called a personal online data store (pod). Pods can be deployed on personal servers or on public servers by other pod providers. Application data in Solid are stored in documents that are identified by a Uniform Resource Identifier (URI) [17].

Pod refers to a personal online data store that is used to store user data on a Solid platform. It is possible for a user to have more than one pod [18]. A user can select diverse pod providers because Solid applications can operate with any pod server without the limitation of service provider and location. Diverse pod providers can provide diverse degrees of security, availability, and reliability. Different pods can access their data resources and deliver content to each other [17].
In summary, Solid is a web decentralization technology, and a Solid platform can provide a relatively effective and safe means of executing web applications. However, Solid technology still mainly focuses on social media networks. Few studies have considered using Solid in the transportation field. To fill this gap, this study applied Solid technology to public transportation data sharing based on the semantic web, and a new alternative approach was proposed to integrate and share these heterogeneous data.

3. Methodology

The methodology used in this study, presented in Figure 1, analyzed the system functional and non-functional requirements by analyzing the capability and usability of diverse public transportation apps. Next, the system design is given, and the system is implemented based on the system requirements and data processing. Finally, recommendations were devised through experiments and analysis of results.

3.1. Requirement Analysis Based on Comparison of Capability and Usability of Public Transportation Apps. The system requirements were collected based on the capability of diverse public transportation apps compared with the usability of diverse public transportation apps. The public transportation apps in this study were categorized into four groups based on their capabilities and usability. The first group includes minicab apps, including Uber and Aqua cars. The second group includes train and coach apps including Trainline, Virgin Trains, and National Express and bus apps such as First Bus and UK Bus. The third group includes car parking apps such as JustPark and YourParkingSpace, and the last group includes Google and the system. The capabilities and usability of these apps are compared below (see Tables 1 and 2, respectively).

Defining attributes of usability, saving favorite routes, and recent searches and routes, as shown in Table 2, belong to an efficiency attribute because they are related to the accuracy and completeness with which users achieve goals. Sending the selected route to the phone, suggesting an appropriate route based on what the user needs, and sharing an appropriate route to car parking is part of a satisfaction attribute because they have a positive effect on the app usage. Help and FAQ functions make it easy to learn to use apps so that the user can rapidly start getting work done through the app, which is the learnability attribute. Social capabilities allow users to share experiences, including comments, pictures, and videos, and interact with other tourists. It is interesting for users to improve the use of a recommender during a visit to a particular place [19].

3.2. Functional and Non-Functional Requirements. User requirements can be divided into two types: functional and non-functional requirements [20]. Functional requirements can be elicited directly from a user through software feature requests [21]. The non-functional requirement can be defined as part of the attributes of the system [20]. The functional requirements of the new system are listed in Table 3 (functional requirement table), and the non-functional requirements of the new system are listed in Table 4 (non-functional requirement table).

4. System Design

4.1. Proposed Data Processing Method. The proposed data processing method has six key elements: data collected, data integrated and generated, data stored, data searched, and data shared. These elements collect data from different data resources and transfer a uniform RDF model and then save the pod server on the Solid platform, and finally share these data through a web application (see Figure 2).

First, data are collected; the public transportation data are collected from tracking devices and sensors that are located in the entire city and installed on cameras and mobile phones [22]. These public data are referred to as open data because they can be freely used and redistributed by someone either for or at marginal cost through existing web applications, for example, data.gov.uk [23]. During the data collection phase, diverse data sources will be collected through different data sources, such as the data of the Uber system [24], the data of the National Express system [25], and the data of car parking systems [26]. Second, we discuss data integrated and generated elements. There are diverse open data sources in public transportation, for example, National Express is XML; Uber and car parking datasets are structured data like CSV; and Trainline data are unstructured data, like TXT. Therefore, it is necessary to consider heterogeneous datasets and translate them into a uniform data type. An appropriate semantic model can supply an interoperable representation of data [27]. Therefore, semantic web technologies are suggested to address these requirements because they will supply the necessary capabilities for public transportation dataset unification of different open datasets. Each data source needed a diverse method to be extracted and changed into a uniform RDF data model because the RDF data model makes it easier to integrate system data than traditional data models, for example, relational data [28].

Third, data are stored; RDF is a graphics-based data model that can represent any data structure. RDFS can be defined within the RDF so that data structures and types can be adapted as the application demands. RDFS and OWL are programs that change diverse data types into a uniform RDF, adding a semantic mark-up that describes the meaning of each data item. Different data were unified in the RDF and stored in the pod server on the Solid platform, which is a new method because Solid technology was first used in the public transportation area. Solid technology allows users to have full control of their data, including access control and storage location.

Fourth is the data searched element: there are two possible methods for data searching on a Solid platform. One is the RESTful method in terms of the LDP [29]. Another is the SPARQL query method [18]. In a Solid platform, all pod servers must support the LDP, while some servers may optionally support SPARQL.
Finally, we have the data shared element: based on the above steps, users will have access to these data based on their different authority levels, such as access to the entire dataset or only part of the dataset and access to modifying data and comments after data are shared on the website.

4.2. System Framework. Figure 2 describes a complete process in which data is collected, integrated, stored, and shared. Each functional capability is defined as a separate function, permitting separate programming and evolution, as shown in Figure 2, which derives from architectures in Slogger: a profiling and analysis system based on semantic web technologies [30]. Figure 3 shows the system framework for implementing a web-based data-sharing system, including transformers, semantic web query algorithms, and visualization technology. This new proposed system framework for implementing public transportation data sharing is mainly based on Solid.

5. System Implementation

5.1. C2RMF (CSV to RDF Mapping File) Algorithm. To change the CSV data of car parking into RDF for integrating heterogeneous data, the new C2RMF (CSV to RDF Mapping File) algorithm was developed. Converting CSV files to RDF files based on mapping files is currently the main way including the C2RMF algorithm. Some existing tools or ways are accomplished for CSV to RDF including our proposed algorithm. But these tools use different mapping techniques, and it is difficult to use and share these mapping engines. Furthermore, most of them lack the use of semantic web technology and W3C recommended standards [31].

C2RMF algorithm is fully in accordance with W3C recommendation [32], a Java-based method which converts CSV data into RDF. If no specific format is provided, the result is serialized as a Turtle file by default. The algorithm can convert CSV files into RDF files and also can change some structured data of public transportation into RDF, adding a semantic mark-up that describes the meaning of each data item. The algorithm can convert each row of the input CSV data into a new instance of a uniform RDF class.

Each value in the column of the source CSV is transformed into a new triple where each key represents a column position in the source, CSV stands for the subject, each property depending on the name of the column header stands for the predicate, and each value of the column stands for the object. The algorithm is also entirely customizable to meet specific user requirements in terms of the mapping file.

Define the RDF graph. Let M and N be a finite set of uniform resource identifiers and literals. A tuple \((s, p, o) \in M \times M \times (M \cup N)\) can be called an RDF triple. Each RDF triple \(t = (s, p, o)\) indicates that resource \(s\) and resource \(o\) have a relationship \(p\), where \(s\), \(p\), and \(o\) represent a subject, a predicate, and an object, respectively, and thus a finite set of triples is called an RDF graph. C2RMF algorithm is shown in Table 5.

Figure 4 shows the transfer process of the C2RMF algorithm and illustrates the corresponding relationship between the contents of the CSV and RDF files.

5.2. Semantic Web Query Process. To retrieve transportation route alternatives from the Solid server, semantic web queries were developed using SPARQL. SPARQL is the W3C standard for creating, querying, and updating linked databases. SPARQL is the standard query language for the RDF model. SPARQL has proven to be a powerful querying language. SPARQL queries in Solid are divided into two queries: local queries and link-following queries. The local query can access only predicates that are located on the local user’s pod, whereas the link-following query can access predicates on many pods [17]. The query process retrieves transportation route alternatives in terms of routeId or tripId. The semantic web query considers the synonym heterogeneity for querying between two stations because the station names are diverse in the domain ontology with the same separate definitions.

Similar to SQL, SPARQL retrieves data from the query dataset through a Select statement to determine which result of the selected data will be returned. Additionally, SPARQL uses a Where clause to state criteria to discover a match in the query dataset. A SPARQL query comprises five parts: namespace prefix, result set, dataset, query triple pattern, and modifiers. Similar to SQL, Running a SPARQL query will return all the match data. In the SPARQL query example
Table 1: Capability of diverse public transportation apps (Y denotes yes, N denotes no, and P denotes partial).

|                          | Search route based on leaving date and depart time | Search route based on starting position and destination | Display the diverse routes including different vehicles | Display the diverse routes including stopped stations | Display ticket and payment information based on searched route | Display schedule based on search route and vehicle | Book user schedule based on user needs | Display traffic accidents of the selected route | Display traffic jam of the selected route | Search car parking based on date and parking close to where user needs it | Create account and sign in |
|--------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|
| Uber                     | Y                               | Y                               | P                               | P                               | P                               | Y                               | Y                               | Y                               | Y                               | Y                               | N                               |
| Aqua cars                | Y                               | Y                               | P                               | P                               | P                               | Y                               | Y                               | Y                               | Y                               | Y                               | N                               |
| National express         | Y                               | Y                               | P                               | Y                               | Y                               | Y                               | Y                               | Y                               | N                               | N                               | N                               |
| First Bus                | Y                               | Y                               | P                               | Y                               | Y                               | Y                               | N                               | N                               | N                               | N                               | N                               |
| UK Bus                   | Y                               | Y                               | P                               | Y                               | Y                               | Y                               | Y                               | N                               | N                               | N                               | N                               |
| Trainline                | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | N                               | N                               | N                               | N                               | N                               |
| Virgin Trains            | Y                               | Y                               | P                               | Y                               | Y                               | Y                               | N                               | N                               | N                               | N                               | N                               |
| YourParkingSpace         | N                               | N                               | N                               | N                               | N                               | N                               | N                               | N                               | N                               | Y                               | Y                               |
| JustPark                 | N                               | N                               | N                               | N                               | N                               | N                               | N                               | N                               | N                               | Y                               | Y                               |
| Google                   | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               |
| My system                | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               | Y                               |
Table 2: Usability of diverse public transportation apps.

|          | Save favorite routes | Save recent searches and routes | Send the selected route to the phone | Suggest an appropriate route based on user needs | Suggest an appropriate route to go to car parking | Help and FAQ | Social aspects |
|----------|----------------------|-------------------------------|-------------------------------------|-----------------------------------------------|-----------------------------------------------|-------------|---------------|
| Uber     | Y                    | Y                             | N                                   | Y                                             | N                                             | Y           | N            |
| Aqua cars| Y                    | Y                             | N                                   | Y                                             | N                                             | Y           | Y            |
| National express | Y               | Y                             | N                                   | Y                                             | N                                             | Y           | Y            |
| First Bus | Y                    | Y                             | N                                   | Y                                             | N                                             | Y           | N            |
| UK Bus   | Y                    | Y                             | N                                   | Y                                             | N                                             | Y           | N            |
| Trainline| Y                    | Y                             | N                                   | Y                                             | N                                             | Y           | N            |
| Virgin Trains | Y         | Y                             | N                                   | Y                                             | N                                             | Y           | N            |
| YourParkingSpace | N         | N                             | N                                   | N                                             | Y                                             | Y           | Y            |
| JustPark | N                    | N                             | N                                   | N                                             | Y                                             | Y           | Y            |
| Google   | Y                    | Y                             | Y                                   | Y                                             | Y                                             | Y           | N            |
| My system | Y                    | Y                             | N                                   | Y                                             | Y                                             | Y           | Y            |

Table 3: Functional requirement.

| No | Description                                                                 | Proposer   |
|----|-----------------------------------------------------------------------------|------------|
| 1  | Search route based on leaving date and depart time and starting position and destination | User       |
| 2  | Display the diverse routes that include different vehicles and detailed stop stations | User       |
| 3  | Display the traffic accident of searching routes                              | User       |
| 4  | Display the traffic jam of searching routes                                   | User       |
| 5  | Suggest an appropriate route based on user needs                              | System     |
| 6  | Search car parking based parking date and parking close to where user needs it through users’ position | User       |
| 7  | Create account and sign in, and find an appropriate route based on user needs and book it | User       |
| 8  | Save favorite routes and recent searching routes                              | User       |
| 9  | Help and FAQ                                                                  | User       |
| 10 | Social media including Twitter, Instagram, and Facebook                       | User       |
| 11 | Suggest an appropriate route to go to car parking                             | System     |
| 12 | The system can restrict different levels of users so only authorized users can do what they ought to do | System     |

Table 4: Non-functional requirement.

| No  | Definition                        | Description                                                                 |
|-----|-----------------------------------|-----------------------------------------------------------------------------|
| 1   | Usability                         | System states that all menus and navigation are easy to use, system can be used by adult members of the public without training |
| 2   | Security                          | System data are stored safely and protected                                  |
| 3   | Compatibility                     | Different browser compatibility and operating system compatibility            |
| 4   | Performance                       | All pages on website system load in less than 5 s                            |
| 5   | Extensibility                     | System architecture can integrate other new public transportation data       |
| 6   | Availability                      | System shall be available for use within 24 hours a day                      |

Figure 2: Data processing graph.
5.3. Visualization. There are two key modules during the visualization component in the web-based data-sharing system visualization. Google Maps can establish a stable connection from the server to the client and provide downloading of extra map information for displaying map information on the client [33]. Additionally, the application programming interface (API) function, provided by Google, comprises a couple of classes, functions, and data structures that could be used by a developer through JavaScript or others [34]. In this system, the Google Map API is called to initialize the selected map area for display on the web page.

5.3.1. SPARQL Support. SPARQL is used to retrieve relative route data through the pod server, and then CSS and JavaScript are used to show selected Uber and bus routes, displaying the message on the map based on retrieving data. Figure 6 shows the visualization framework for implementing a web-based data-sharing system.

6. Experiment

The experiment can be divided into two parts, the performance test of the C2RMF algorithm and system testing.

6.1. Conversion Performance Test. To verify the performance of the C2RMF algorithm, two ways of converting CSV to RDF data were tested in this testing process, one is the C2RMF algorithm, and the other is a transformer tool named stlab.csv2rdf-1. This csv2rdf is a typical Java-based and open-source tool, which depends on Apache Jena to convert CSV data into RDF [35]. There are other transfer tools that can achieve this purpose, such as Geometry RDF and Table 6. But Csv2rdf is chosen as it is very easy and intuitive to use, and well provide to achieve data transformation.

Table 5: C2RMF algorithm.

| Input: CSV files; mapping files. | Output: RDF file |
|--------------------------------|------------------|
| (1): Read the CSV file to get the column size | (1): RDF file |
| (2): Read the CSV file to get the row size | (2): RDF file |
| (3): Get column. Property of the CSV file | (3): RDF file |
| (4): For i = 1, . . . , Rowsize do | (4): RDF file |
| (5): For each Column [i]. Property do | (5): RDF file |
| (6): s ← i | (6): RDF file |
| (7): p ← Get mapping. URI [i] of the mapping file | (7): RDF file |
| (8): o ← Columns [i]. value | (8): RDF file |
| (9): Generate RDF triple (s, p, o) | (9): RDF file |
| (10): End for | (10): RDF file |
| (11): Return RDF triple in the RDF datasets. | (11): RDF file |

Figure 2: System framework.
execution time of the C2RMF algorithm and the csv2rdf based on these four datasets, and the four stages of experimental results are listed in Figure 7. The other refers to the two ways of CSV Transformers RDF based on Metro_Interstate_Traffic_dataset, and these experimental results are listed in Figure 8.
6.2. System Testing. This part of system testing mainly includes functional testing and non-functional testing.

6.2.1. Testing Environment. The system hardware testing environment includes a desktop computer configured with an Intel (R) Core(TM) i7-4702MQ CPU 2.2 GHz and 8 GB of RAM. The system software testing environment comprises a Windows 10 Professional 64 byte operating system, a local Solid server version 5.1.6, SpringBoot version 2.1.6, an embedded Tomcat server, which is considered a local web server, and a transformer tool named CSV Transformers RDF algorithm.

6.2.2. Testing Data. During the test process, diverse CSV data were collected from different data sources using diverse addressing mechanisms [24–26].

6.2.3. Functional Testing and Results. Functional testing was used for an achieved program. The aim was to demonstrate that it supplies all of the behaviors required of it. The option of test cases is in terms of the user requirement of the software entity under test [37]. Functional testing is a form of black-box testing. Based on the above, functional testing was considered the main testing technology during the testing phase.

These figures show the functional testing results based on the testing data. Figure 9 shows the bus route list for sharing data on the Solid platform. Figure 10 shows a bus route on the Google Maps and retrieves data from the Solid platform. Figure 11 shows an Uber route on the Google Maps and retrieves data from the Solid platform. Figure 12 shows a car parking position on the Google Maps and retrieves data from the Solid platform.

Based on Table 7, the functional testing results of the new system are listed in Table 8.

6.2.4. Non-Functional Testing and Results. According to Table 4, the non-functional testing results are listed below:
Figure 9: The bus route list through Solid.

Figure 10: A bus route.
Figure 11: Uber route.

Figure 12: A car parking position.
6.3. Experiment Summary. In total, approximately 80 typical test cases were executed and presented in this paper. Each test case is designed to produce useful data. These test cases were chosen to develop data conversion, data storage, data retrieval, and visualization. Additionally, it is necessary to verify the user requirements and non-functional requirements of these test cases. In this section, some typical examples of these tests are presented, such as converting a heterogeneous data source into uniform RDF data, storing RDF data in the pod server to retrieve data through SPARQL, and finally displaying data on the web page based on Google Maps, CSS, and JavaScript. The test results demonstrate that the new website can meet user requirements.

Furthermore, applications are implemented as client-side web, which reads and writes data directly from the pods in this website in terms of a Solid platform. Multiple applications can also reuse the same data on pod servers. In addition, the new website system sustained the view that sharing public transportation data services could be improved through semantic web and Solid technologies. The website was also developed that is free and provides several protocols, for example, SPARQL. It is efficient for developing applications through Solid platforms.

In addition, the new system is a well-established standard for publishing and managing unstructured or structured data on the web, gathering and bridging knowledge from different data sources. C2RMF algorithm was developed to convert CSV into RDF, and also its performance test was verified.

6.4. Experiment Analysis. Figure 7 shows the comparison of the execution time of the C2RMF algorithm and the csv2rdf on different datasets. Figure 8 shows the comparison of the execution time of the C2RMF algorithm and the csv2rdf based on the different number of rows. In terms of Figures 7 and 8, we can make the following observations:

(i) The C2RMF algorithm and csv2rdf had the highest conversion execution time for the test dataset, Metro_Interstate_Traffic_dataset, 2768 kb, while another test dataset, bike-sharing-day-dataset, with 56 kb, had the least running time, as the execution time usually depends on the properties of the datasets.

(ii) The execution time of both the C2RMF algorithm and the csv2rdf exhibits approximately a linear growth rate as the number of dataset rows. As shown, when the number of dataset rows is low, then the execution time is less; it gradually increases based on the properties and sizes of the datasets.
On the other hand, Figure 9 shows the alternative bus route list to arrive at the destination based on this CSV file of the National bus system, which aims to provide the best alternative routes to arrive at the destination in urban areas according to open CSV data. Figure 10 shows a further test case; according to Figure 9, it retrieved each bus station of the selected bus route through SPARQL and finally displayed them on the website through the Solid platform. Figure 11 shows a test case from the CSV file of the Uber system. It suggests that these alternative routes arrive at the destination. The processing flow is similar to that in Figure 10, and the route is also shown on the website with Google Maps. Figure 12 shows the test case from a car parking CSV file, which suggests suitable car parking locations for a car based on its location and is shown on the web page with Google Maps.

However, there are some limitations:

(i) The data source is a single type file: among the different data sources, only the CSV file was selected and converted into an FDF file through our developed converting algorithm during the testing process. In a real environment, other data source files must be selected and converted.

(ii) History data source: the different open data sources were history data and not real-time data. It is helpful to provide online data in a real environment.

(iii) Lack of SPARQL interface: website developers need to be experienced with diverse data schemas and query evaluations to solve effective SPARQL queries.

7. Conclusion and Recommendations

In this paper, we proposed a novel approach that mainly included the proposed data processing method, the new C2RMF algorithm, the proposed system framework, and the web-based data-sharing system. The approach can achieve to integrate and share heterogeneous public transportation data to provide personalized sharing of these data between different users. The research results on the publicly available datasets demonstrate the significance of the approach from two aspects. (1) The proposed data processing method involves the management process of the full life cycle of data including data collected, data integrated and generated, data stored, data searched, and data shared. (2) The proposed approach provided a unified data operation mode for information storage and retrieval of heterogeneous public transportation data. It is useful to manage these data.

In future work, it would be very interesting to do the following. (1) We plan to convert other data types of public transportation including structured data into RDF data through our C2RMF algorithm. (2) We will try to forecast short-term traffic flow based on these public transportation data.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by Major Public Welfare Projects of Henan Province (201300210500).

References

[1] T. M. O’Brien, “Vivek Kundra: federal cio in his own words,” 2011, http://radar.oreilly.com/2009/03/vivek-kundra-federal-cio-in-hi.html.2011.
[2] M. Maksimovic, “The role of green internet of things (G-IoT) and big data in making cities smarter, safer and more sustainable,” International Journal of Computing and Digital Systems, vol. 6, no. 4, pp. 175–184, 2017.
[3] X. Ma, K. Zhang, L. Zhang et al., “Data-Driven niching differential evolution with adaptive parameters control for history matching and uncertainty quantification,” SPE Journal, vol. 26, no. 02, pp. 993–1010, 2021.
[4] G. Sun, C. Li, and L. Deng, “An adaptive regeneration framework based on search space adjustment for differential evolution,” Neural Computing & Applications, vol. 33, no. 15, pp. 9503–9519, 2021.
[5] L. Ding, S. Li, H. Gao, Y. J. Liu, L. Huang, and Z. Deng, “Adaptive neural network-based finite-time online optimal tracking control of the nonlinear system with dead zone,” IEEE Transactions on Cybernetics, vol. 51, no. 1, pp. 382–392, 2021.
[6] J. Yang, M. Xi, B. Jiang, J. Man, Q. Meng, and B. Li, “FADN: fully connected attention detection network based on industrial video,” IEEE Transactions on Industrial Informatics, vol. 17, no. 3, pp. 2011–2020, 2021.
[7] S. Wang, W. A. Higashino, M. A. Hayes, and M. A. M. Capretz, “Service evolution patterns,” in Proceedings of the 2014 IEEE International Conference on Web Services, ICWS, 2014, pp. 201–208, Anchorage, AK, USA, June 2014.
[8] T. Espinha, A. Zaidman, and H. G. Gross, “Web API growing pains: loosely coupled yet strongly tied,” Journal of Systems and Software, vol. 100, pp. 27–43, 2015.
[9] J. Samuel and C. Rey, “Challenges in integrating multiple heterogeneous and autonomous web services using mediation approach,” in Proceedings of the 2016 Eleventh International Conference on Digital Information Management (ICDIM), pp. 185–190, IEEE, Porto, Portugal, September 2017.
[10] R. Tan, R. Chirkova, V. Gadepally et al., “Enabling query processing across heterogeneous data models: a survey,” in *Proceedings of the 2017 IEEE Int'l Conf. on Big Data (Big Data)*, pp. 3211–3220, IEEE, Boston, MA, USA, December 2017.

[11] J. Domingue, D. Fensel, and J. A. Hendler, “Introduction to the semantic web technologies,” in *Handbook of Semantic Web Technologies*, D. John, F. Dieter, and J. A. Hendler, Eds., pp. 3–41, Springer-Verlag, Berlin, Heidelberg, 2011.

[12] C. Bizer, T. Heath, and T. Berners-Lee, “Linked data - the story so far,” *International Journal on Semantic Web and Information Systems*, vol. 5, no. 3, pp. 1–22, 2009.

[13] Z. Wei, *Web-based Data Sharing System for Public+ Transportation*, Unpublished Master’s Thesis, University of Portsmouth, ProQuest Dissertations and Theses Global, Portsmouth, England, 2019.

[14] "RDF vocabulary description language 1.0: RDF schema," 5 November 2007, http://www.w3.org/2004/REC-rdf-schema-20040210.

[15] T. Berners-Lee and E. Miller, "The semantic web," 2002, http://www.w3.org/2002/Talks/01-sweb/.

[16] M. Dean, D. Connolly, F. van Harmelen et al., "OWL web ontology language 1.0 Reference," July 2002, http://www.w3.org/TR/owl-ref/.

[17] "Optimizations over decentralized RDF graphs," in *Proceedings of the IEEE International Conference on Data Engineering, IEEE Computer Society*, pp. 139–142, San Diego, CA, USA, April 2017.

[18] E. Mansour, A. Vlad Sambra, S. Hawke et al., "A demonstration of the solid platform for social web application," in *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 223–226, Montréal, Canada, April 2016.

[19] J. Borras, A. Moreno, and A. Valls, "Intelligent tourism recommender systems: a survey," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7370–7389, 2014.

[20] D. Leffingwell and W. Don, *Managing Software Requirements: A Use Case Approach*, pp. 978–0321122476, Addison Wesley, Boston, MA, USA, 2003.

[21] E. Guzman and W. Maalej, "How do users like this feature? a fine grained sentiment analysis of app reviews," in *Proceedings of the 2014 IEEE 22nd International Requirements Engineering Conference (RE)*, pp. 153–162, IEEE, Karlskrona, Sweden, August 2014.

[22] M. Gohar, M. Muzammal, A. U. Rahman, and T. S. S. Smart, "Smart tss: defining transportation system behavior using big data analytics in smart cities," *Sustainable Cities and Society*, vol. 41, pp. 114–119, 2018.

[23] "How linked data is transforming eGovernment," January 2013, https://www.slideshare.net/mlout/d432-case-studydataintegrationv015.

[24] "Uber Pickups in NYC, dataset by data-society," 2015, https://data.world/data-society/uber-pickups-in-ny.

[25] "National express data," 2019, http://www.basemap.co.uk/data/NCSD/NCSZC.ZIP.

[26] "Council car parks, data.gov.uk," 2019, https://data.gov.uk/dataset/42589d42769/council-car-parks.

[27] R. Uceda-Sosa, B. Srivastava, and R. J. Schloss, "Building a highly consumable semantic model for smarter cities," in *Proceedings of the AI for an Intelligent Planet*, pp. 1–8, Barcelona Spain, July 2011.