International Journal of Geo-Information

MDPI

Article

Algebraic Operations on Spatiotemporal Data Based on RDF

Lin Zhu, Nan Li and Luyi Bai*

School of Computer and Communication Engineering, Northeastern University (Qinhuangdao), Qinhuangdao 066004, China; zhulin@neuq.edu.cn (L.Z.); lishuchaojishuai@126.com (N.L.)

* Correspondence: baily@neuq.edu.cn; Tel.: +86-135-1335-0400

Received: 10 December 2019; Accepted: 27 January 2020; Published: 30 January 2020

Abstract: In the context of the Semantic Web, the Resource Description Framework (RDF), a language proposed by W3C, has been used for conceptual description, data modeling, and data querying. The algebraic approach has been proven to be an effective way to process queries, and algebraic operations in RDF have been investigated extensively. However, the study of spatiotemporal RDF algebra has just started and still needs further attention. This paper aims to explore an algebraic operational framework to represent the content of spatiotemporal data and support RDF graphs. To accomplish our study, we defined a spatiotemporal data model based on RDF. On this basis, the spatiotemporal semantics and the spatiotemporal algebraic operations were investigated. We defined five types of graph algebras, and, in particular, the filter operation can filter the spatiotemporal graphs using a graph pattern. Besides this, we put forward a spatiotemporal RDF syntax specification to help users browse, query, and reason with spatiotemporal RDF graphs. The syntax specification illustrates the filter rules, which contribute to capturing the spatiotemporal RDF semantics and provide a number of advanced functions for building data queries.

Keywords: graph algebra; spatiotemporal RDF; syntax specification

1. Introduction

The Resource Description Framework (RDF) [1] was originally designed as a metadata model to publish or exchange data on the Semantic Web, and has since become the W3C standard. Recently, it has been used as a general method for conceptual description, and some researchers have started to focus on spatiotemporal data modeling and algebraic operations.

Towards spatiotemporal data modeling, there have been some achievements in representing spatiotemporal entities, such as a temporal data model [2,3], a spatial data model [4], and a spatiotemporal data model [5]. In previous studies on temporal data modeling, Tappolet et al. [6] present a syntax and storage format based on named graphs to express a temporal RDF. Due to the fact that all entities can be linked with relations through labels, Hernández et al. [7] put forward an approach that adds temporal qualifiers and values to the RDF model representation in the form of labels. In this approach, the RDF triples are expanded into a five-tuple (s, p, o, q, v), where (s, p, o) refers to the primary relation, q is the temporal qualifier property, and v is the temporal qualifier value. Besides this, there are some models that add temporal labels to RDF triples to form quads [8,9]. In addition to temporal models, there are some studies about spatial RDF modeling. For example, GeoRDF [10] is an RDF-compatible profile for describing geometric information (points, lines, and polygons). In GeoRDF, GeoMetadataOverSvg is the geographic information notation, which plays a significant role in the spatial data connection of the Semantic Web. Benefiting from that, GeoRDF can be used to represent any point on the Earth. Regarding spatiotemporal data, the structured

spatiotemporal RDF was first proposed by Koubarakis et al. [11]. It develops the stRDF data model based on RDF and puts forward the stSPARQL query language. The contribution of stRDF is regulating the representation principle of spatiotemporal data in RDF and standardizing the spatiotemporal data querying. It can also be applied to several spatiotemporal-related applications [12,13,14]. However, the spatial information fails to associate with temporal information in the stRDF model, which means that it has a weak ability to record dynamically changing data. When the knowledge graph is updating, the changes in spatiotemporal attribute values cannot be captured in time and lead to inconsistencies in the data. Moreover, it will probably return multiple results or errors when we query it for the spatial information at a given time.

With the prompt development of RDF graph querying [6,15], graph algebra has received widespread attention. It is the key to applying standard database-style optimization to queries. In previous studies, several different RDF algebras have been proposed by research groups coming from academia. Due to the lack of an RDF algebra, such query languages use APIs to describe their semantics, and optimization issues have been mostly neglected. To solve this problem, Frasincar et al. [16] propose RAL (an RDF algebra) as a reference mathematical study for RDF query languages. Robertson [17] proposes an algebraic operation of triadic relations for the RDF. An important aspect of this algebra is an encoding of triples, which implements a kind of reification. Chen et al. [18] incorporate semantics that inference into query answering and propose an RDF algebra based on a layered RDF graph model. The matrix algebra method MAGiQ was put forward by Jamour et al. [19]. MAGiQ represents an RDF graph as a sparse matrix, and translates SPARQL queries to matrix algebra programs, which takes advantage of the existing software infrastructure to process sparse matrices and is optimized for many architectures (e.g., central processing units (CPUs), graphical processing units (GPUs), and distributed architectures) effortlessly. On the basis of previous work, Thakkar et al. [20] consolidate existing graph algebra operators from the literature and propose two new traversal operators in an integrated graph algebra. Although researchers have proposed a lot of algebraic methods to assist with RDF queries, these methods or models cannot support spatiotemporal RDF queries. Therefore, it is necessary to develop a relatively complete spatiotemporal RDF graph algebra.

Motivated by such an observation, in this paper, we study algebraic operations and a spatiotemporal syntax specification based on the RDF. We: (*i*) present a general algebraic operational framework for manipulating spatiotemporal data, which can represent the content of data more clearly and support spatiotemporal RDF graphs specifically; and (*ii*) propose a spatiotemporal RDF syntax specification. It will be of great benefit to query explicit spatiotemporal RDF data and capture spatiotemporal RDF semantics.

The remainder of this paper is organized as follows. Section 2 presents related work. In Section 3, spatiotemporal algebraic operations based on the RDF are proposed. Section 4 explores the spatiotemporal RDF syntax specification, and Section 5 presents our conclusions and future work.

2. Related Work

As it contributes a lot to optimizing queries, algebra has been identified as a core operation in querying. The research results that are presented in this section mainly include works about spatiotemporal algebra and the algebra of RDF graphs.

2.1. Algebra for Spatiotemporal Data

With the emergence of a large amount of spatiotemporal data, researchers began to devote themselves to the study of spatiotemporal algebra. To support spatial features, Córcoles et al. [21] present the Geography Markup Language (GML) based on XML. It inherits the interoperability of the XML and allows for the exchange of geographic information on the Web, and an extension to algebraic operations is provided on the basis of GML and a query language. Towards temporal data, Pan et al. [22] represent temporal aggregates in OWL-Time, and propose a systematic way of mapping iCalendar recurrence sets to OWL-Time temporal sequences. Baratis et al. [23] propose a four-dimensional (4D)-Fluent representation that represents temporal information in OWL, where

concepts varying in time are represented as 4D objects, and put forward the TOQL query language. Batsakis et al. [24] enhance this approach with qualitative temporal expressions allowing for the representation of temporal intervals with unknown starting and ending points by means of their relation (e.g., "before" and "after") to other time intervals. To handle qualitative temporal relationships and the extended 4D-Fluent representation, they also extend the TOQL query language. In [25], Moffitt et al. combine advances in graph databases and temporal relational databases. Then, they propose an algebra called TGA that adheres to point-based semantics. TGA includes principled temporal generalizations of conventional graph operators as well as novel operators that support exploratory analysis of evolving graphs at different levels of temporal and structural granularity. Aiming to perform operations over spatiotemporal data, Perry et al. [26] study geospatial and temporal semantic analytics and Hakimpour et al. [27] present practical approaches to data processing in the space, time, and theme dimensions using existing Semantic Web technologies. Perry et al. [28] devise SPARQL-ST to support spatiotemporal queries based on SPARQL. Bai et al. [29] develop an algebra based on Native XML to manipulate spatiotemporal XML data. In this work, the logical structure of a spatiotemporal database, data type systems, and querying operations were investigated. Bai et al. [30] deal with fuzzy information and propose an algebra for fuzzy spatiotemporal data in XML.

2.2. Algebra of RDF Graphs

There is a rich set of studies on RDF graph algebra. For instance, an algebraic operation of triadic relations for RDF is introduced in [17]. Jamour et al. [19] propose a matrix algebra method to answer RDF graph queries. Frasincar et al. [16] propose a prototype of an RDF algebra, including three types of operations: the first is the extraction operation, the second is the loop operation, and the third is the model construction operation. However, RAL does not support RDF graph structure queries. Chen et al. [18] introduce a set of operations for manipulating RDF graphs and an RDF query algebra (LAGAR) is proposed. LAGAR includes four operations: pattern-matching operations, construction operations, graphics set operations, and related functional operations, and this algebra optimizes RDF graph model matching operations. The use of a relational algebra for SPARQL query processing is investigated in [31], in which the transformation from SPARQL into an abstract relational algebra is presented and the differences between the semantics of SPARQL and that of the relational model are discussed. GeoSPARQL [32] attempts to unify data access for the geospatial Semantic Web. Battle et al. [33,34] describe the motivation for GeoSPARQL and the implementation of GeoSPARQL. In order to better manage fuzzy data, Zuo et al. [35] introduce a picture fuzzy graph based on a picture fuzzy relation, and describe the utility of the picture fuzzy graph. Ma et al. [36] put forward a fuzzy RDF model. In addition, a fuzzy RDF algebra is formally proposed and a set of algebraic operations is developed. To rewrite algebra expressions in a form that satisfies certain needs, they also present some algebraic equivalences based on a data graph isomorphism. Because the interval type 2 fuzzy set (IT2FS) increases the number of degrees of freedom to express uncertainty in the edge weight and has a greater capacity to describe fuzzy information in a logically correct manner, Dey et al. [37] propose the minimum spanning tree problem with an undirected connected weighted interval type 2 fuzzy graph (FMST-IT2FS), which can be used to optimize the query of fuzzy RDF graphs. Besides this, several RDF graph algebras are proposed to deal with specific domains, such as querying a large-scale distributed triple-store system [38] and optimizing RDF graph pattern matching in MapReduce [39].

In recent years, some query methods based on the spatiotemporal RDF have been developed, including stSPARQL [11], SPARQL-ST [28], gst-store [40], and ST-SPARQL [41]. Koubarakis et al. [11] propose the query language stSPARQL, which is an extension of SPARQL. Compared with SPARQL, it has the ability to query spatiotemporal attributes. The main statements are the *SELECT* statement, the *Filter* statement, and the *Having* statement. Since stRDF cannot dynamically connect spatiotemporal data, it is difficult to use stSPARQL to query large-scale dynamic spatiotemporal data. Perry et al. [28,42] describe a framework built over the RDF metadata model for analysis of thematic, spatial, and temporal relationships between named entities, and then present a formal syntax and

semantics for SPARQL-ST based on the formalization of the SPARQL syntax given by [43]. In addition, Perry et al. [44] also give an overview of GeoSPARQL's implementation in Oracle Spatial and Graph and show how to load, index, and query spatial RDF data. Wang et al. [40,45] present a spatiotemporal-information-integrated RDF data management system called gst-Store and introduce a spatiotemporal query language that extends the SPARQL language with spatiotemporal assertions to query spatiotemporal-information-integrated RDF data. In [41], the authors put forward a spatiotemporal data type that adds spatial data on Geo-ontology to temporal data. They suggest that it should be applied in ST-OW because using the ST-ontology, which has been added to the temporal data in the semantic Web service, will make available new information for various inferences and queries.

However, the algebraic operations of the abovementioned RDF graph are mainly used for the traditional RDF dataset, which cannot handle a complex spatiotemporal RDF dataset. Therefore, this paper aims to extend the traditional RDF graph operation and establish an RDF graph algebra for spatiotemporal data.

3. Spatiotemporal RDF Semantics and Graph Algebra

In this section, we explore the spatiotemporal RDF semantics and graph algebra of spatiotemporal knowledge graphs.

3.1. Spatiotemporal RDF Semantics

In the following, we further study the semantics of the spatiotemporal RDF. Before that, we propose a spatiotemporal RDF model called stRDFS. stRDFS is defined as follows:

Definition 1. Given a URI set R, an empty vertex set B, a text description set K, a temporal data set I, and a spatial data set S, an stRDFS expression is g (s, p: <t, l>, o), where:

- s is a resource name and $s \in R \cup B$.
- p is a property name and $p \in R$.
- o is a value and $o \in R \cup B \cup K \cup I \cup S$.
- $t \in I$ is temporal data.
- $l \in S$ is spatial data.

In Definition 1, to solve the problem of data inconsistencies in stRDF, we add spatial labels and temporal labels to the predicate to associate spatial data with temporal data to form a spatiotemporal predicate p. When spatiotemporal data change, spatiotemporal attributes associated with them will change as well.

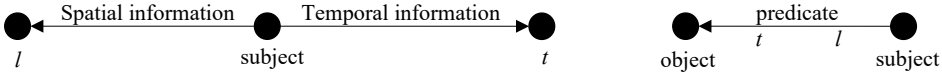
Definition 2. Given an stRDFS expression g (s, p: <t, l>, o), $U = \{f_{s\text{-}o}, f_{s\text{-}t}, f_{s\text{-}l}, f_{p\text{-}t}, f_{p\text{-}l}\}$ is a mapping set of g, the value range of mapping $f_{x\text{-}y}$ is denoted as *Range* ($f_{x\text{-}y}$), *where Range* ($f_{x\text{-}y}$) = y, and an stRDFS graph for g is a labeled graph G (V, E, F, λ, T, L), where:

- $V = s \cup Range$ (U) is the set of vertexes.
- $E = \{(r, r')\}$ is the set of edges from r to r', where $\forall r, r' \in V$.
- F (r, r') = $\{f \mid (r, f$: <t, l>, $r') \in G\}$ is the set of mappings of E, where $\forall r, r' \in V$*.
- λ is the set of labels given by vertexes or edges.
- $T \in Range$ ($f_{s\text{-}t} \cup f_{p\text{-}t}$).
- $L \in Range$ ($f_{s\text{-}l} \cup f_{p\text{-}l}$).

In Definition 2, f is a mapping relationship. $f_{s\text{-}o}$ represents the mapping, whose function is expressed as an attribute name, from s to o. The mapping $f_{s\text{-}t}$ indicates that s is linked with temporal data, and the formed triple is (s, p, t). In (s, p, t), the property represents "temporal information" and the attribute value t represents temporal data. The mapping $f_{s\text{-}l}$ indicates that s is linked with spatial data and the expression is (s, p, l). In the triple, the attribute represents "spatial information" and the attribute value l represents spatial data. The mapping $f_{p\text{-}t}$ indicates that p is linked with temporal data and combines with the other mappings to form an stRDFS tuple. When $f_{p\text{-}t}$ is combined with $f_{s\text{-}o}$, the formed tuple is (s, p: t, o), indicating that the temporal data describe the valid time of (s, p, o). When $f_{p\text{-}t}$ is combined with $f_{s\text{-}t}$, the formed tuple is (s, p: t_2, t_1), indicating that the valid time of s is t_1, and the valid time of the tuple (s, p, t_1) is t_2. When $f_{p\text{-}t}$ is combined with $f_{s\text{-}l}$, a tuple (s, p: t, l) is formed, indicating

that s is linked with spatial data l, and the valid time of tuple (s, p: t, l) is t. The mapping $f_{p\text{-}l}$ represents that p is linked with spatial data and combines with other mappings to form an stRDFS tuple. When $f_{p\text{-}l}$ is combined with $f_{s\text{-}o}$, the formed tuple is (s, p: l, o), indicating that the spatial data l describes (s, p, o). When $f_{p\text{-}l}$ is combined with $f_{s\text{-}t}$, the formed tuple is (s, p: l, t), indicating that the valid time of s is t, and the spatial data of (s, p, t) is l. When $f_{p\text{-}l}$ is combined with $f_{s\text{-}l}$, a tuple (s, p: l_2, l_1) is formed, indicating that s is linked with the spatial data of l_1, and the spatial data of (s, p, l_1) is l_2. The mapping $f_{p\text{-}o}$ is illegal. The $f_{o\text{-}t}$ and $f_{o\text{-}l}$ logically represent the temporal data and spatial data of o, respectively. In the stRDFS structure, $f_{o\text{-}t}$ and $f_{o\text{-}l}$ are converted into $f_{s\text{-}t}$ and $f_{s\text{-}l}$, and they can appear as separate tuple mappings. For instance, (s_1, p, o: t) can be converted into two tuples (s_1, p_1, o) and (s_2, p_2, t), where $s_2 = o$ and p_2 represents "temporal information". Similarly, (s_1, p, o: l) can be converted into two tuples (s_1, p_1, o) and (s_2, p_2, l), where $s_2 = o$ and p_2 represents "spatial information".

According to Definition 2, there are two cases. The first one is that stRDFS graph vertexes contain spatiotemporal information, in this case $T \in Range$ ($f_{s\text{-}t}$) and $L \in Range$ ($f_{s\text{-}l}$), as shown in Figure 1(a). The second case is that the stRDFS graph edges contain spatiotemporal information, in this case $T \in Range$ ($f_{p\text{-}t}$) and $L \in Range$ ($f_{p\text{-}l}$), as shown in Figure 1(b).



(a) $T \in Range$ ($f_{s\text{-}t}$) and $L \in Range$ ($f_{s\text{-}l}$) (b) $T \in Range$ ($f_{p\text{-}t}$) and $L \in Range$ ($f_{p\text{-}l}$)

Figure 1. Representation of spatiotemporal information in an stRDFS graph.

Definition 3. Given two stRDFS graphs G_1 and G_2, the implication relation of G_1 and G_2 is as follows:

- *When G_1, G_2 are stRDFS basis graphs, $G_1 \models G_2$ only if G_1 (Ti) $\models G_2$ (Ti) and G_1 (Si) $\models G_2$ (Si).*
- *When G_1, G_2 are stRDFS graphs, $G_1 \models G_2$ only if μ_1 (G_1) for every base graph instance of G_1, there exists a base graph instance μ_2 (G_2) of G_2 and μ_1 (G_1) $\models \mu_2$ (G_2).*

Here, $\models$ denotes the implication relation and μ represents the mapping of entities to attribute values. The closure of figure G can be expressed as G'. $\cup_{(Si \cup Ti)} G$ ($Si \cup Ti$) is the underlying stRDFS graph of the stRDFS graph G, the union of the graphs G ($Si \cup Ti$), and the subgraphs of G.

Definition 4. Given an stRDFS graph G, the largest set and block closure are defined as follows.

- *G' is the largest set on universe (G) that adds all RDF vocabularies, which can be expressed as tcl (G), where $G \subseteq G'$ or $G \models G'$.*
- *The block closure of G is expressed as scl (G). It is a spatiotemporal RDF graph defined by $\cup_{Si, Ti}$ (cl(G ($Si \cup Ti$)) $^{Si, Ti}$, where cl(G ($Si \cup Ti$) is an arbitrary closure of the RDF graph $G_{(Si \cup Ti)}$.*

Theorem 1. *For two stRDFS graphs G_1 and G_2, a necessary and sufficient condition of $G_1 \models_{(Si \cup Ti)} G_2$ is that there is a mapping from G_2 to scl (G_1).*

Proof of Theorem 1.

Sufficiency. Define μ as a mapping from G_2 to $scl(G_1)$. $\mu_1(G_1)$ is an example of a base graph, such that $\mu_2 = \mu \circ \mu_1$. We can obtain the conclusion that for any temporal function Ti and any spatial function Si, there is $\mu_2 G_2$ ($Si \cup Ti$) $\subseteq \mu_1(cl(G_1))$ ($Si \cup Ti$) and $\forall Si$, $\forall Ti$, $\mu_1(G_1)$ ($Si \cup Ti$) $\models_{(Si \cup Ti)} \mu_2(G_2)$ ($Si \cup Ti$). Therefore, $\mu_1(G_1) \models_{(Si \cup Ti)} \mu_2(G_2)$.

Necessity. Define μ_1 as the mapping from any variable X in G_1 to a different constant C_x. Make $\mu_2(G_2)$ an example of a basic graph, $\forall Si$, $\forall Ti$, $\mu_1(G_1)$ ($Si \cup Ti$) $\models_{(Si \cup Ti)} \mu_2(G_2)$ ($Si \cup Ti$). It is easy to see that $\forall$ Ti, $\forall$ Si, $\mu_2 G_2$ ($Si \cup Ti$) $\subseteq \mu_1(cl\ (G_1))$ ($Si \cup Ti$). $\mu_2 G_2 \subseteq \cup_{(Si \cup Ti)}$ (cl (μ_1 (G_1) ($Si \cup Ti$))) $^{(Si \cup Ti)}$. Therefore, μ_2 is a mapping from G_2 to $scl(G_1)$.

Theorem 2. *Given three stRDFS graphs A (V_A, E_A, F_A, T_A, L_A), B (V_B, E_B, F_B, T_B, L_B), and C (V_C, E_C, F_C, T_C, L_C), an isomorphism from A to B is a bijective function $h: V_A \rightarrow V_B$ and it is an equivalence relation.*

Proof of Theorem 2.

Reflexivity. Consider the identity map $h: V \rightarrow V$ such that $\forall s \in V$, $h(s) = s$. h is a bijective mapping satisfying $\forall s \in V$, $T(s) = T(h(s))$, and $L(s) = L(h(s))$. Hence, h is an isomorphism of the spatiotemporal graph to itself. Therefore, it possesses reflexivity.

Symmetry. Consider the identity mapping $h: V_A \rightarrow V_B$ such that h (s_A) = s_B, $s_A \in V_A$ satisfying T_A (s_A) = T_B ($h(s_A)$) and L_A (s_A) = L_B ($h(s_A)$). As h is bijective by $h(s_A) = s_B$, $s_A \in\in V_A$, then $h^{-1}(s_B)$, $\forall s_B \in V_B$. Therefore, $s_A \in V_A$, T_A (s_A) = T_B ($h(s_A)$), then T_A ($h^{-1}(s_B)$) = T_B (s_B) ($\forall s_B \in V_B$). Additionally, $s_A \in V_A$, $L_A(s_A)$

= $L_B(h(s_A))$, then $L_A(h^{-1}(s_B)) = L_B(s_B)$ ($\forall s_B \in V_B$). Therefore, we obtain h^{-1}: $V_B \to V_A$, which is isomorphic from *B* to *A*.

Transitivity. Suppose that h_1: $V_A \to V_B$ and h_2: $V_B \to V_C$ are isomorphisms of *A* onto *B* and *B* onto *C*, respectively. As h_1 is a bijective map $h_1(s_A) = s_B$, $s_A \in V_A$ satisfies $T_A(s_A) = T_B(h(s_A))$, $\forall s_A \in V_A$, and $L_A(s_A) = L_B(h(s_A))$. In the same way, $h_2(s_B) = s_C$, $s_B \in V_B$ satisfies $T_B(s_B) = T_C(h(s_B))$, $\forall s_B \in V_B$, and $L_B(s_B) = L_C(h(s_B))$. From what has been discussed above, we draw the conclusion that $T_A(s_A) = T_B(s_B) = T_C(s_C)$ and $L_A(s_A) = L_B(s_B) = L_C(s_C)$.

Hence, $h_2 \circ h_1$ is an isomorphism between *A* and *C*. It satisfies transitivity.

In conclusion, the isomorphism between spatiotemporal RDF graphs is an equivalence relation.

3.2. Spatiotemporal Classes and Descriptions in the Spatiotemporal Domain

In order to introduce the stRDFS description more clearly, we introduce several main classes: *strdfs: SpatialObject*, *strdfs: SpatialGeometry*, *strdfs: SpatialFeature*, *strdfs: TemporalObject*, *strdfs: TimeSlice*, *strdfs: TemporalFeature*, *strdfs: SpatiotemporalObject*, *strdfs: SpatiotemporalGeo*, and *strdfs: SemiLinearPointSet.* Their relations are shown in Figure 2.

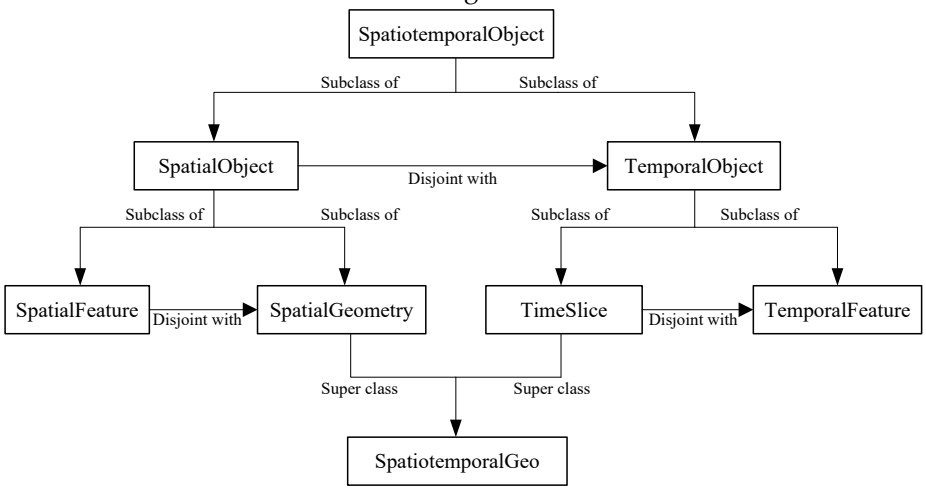


Figure 2 . The relations among the main stRDFS classes.

The class *strdfs: SpatialObject* represents a set of all entities with only spatial information. Its subclass *strdfs: SpatialGeometry* describes parameter *Si* in the stRDFS model (*s*, *p*: <*Ti*, *Si*>, *o*), including the data on latitude, longitude, and altitude, and the other subclass *strdfs: SpatialFeature* describes landform, terrain, and so on. The class *strdfs: TemporalObject* is a set of all entities that contain temporal data. Its subclass *strdfs: TimeSlice* describes parameter *Ti* in the stRDFS model (*s*, *p*: <*Ti*, *Si*>, *o*), and the other subclass *strdfs: TemporalFeature* includes other temporal data on temporal entities, such as a time zone, a tense, a time dimension, or the time of existence. The class *strdfs: SpatiotemporalObject* is a set of all spatiotemporal entities, which is a superset of *strdfs: TemporalObject* and *strdfs: SpatialObject*. The class *strdfs: SpatiotemporalGeo* describes geometric data of spatiotemporal entities and it is a superset of *strdfs*: *SpatialGeometry* and *strdfs*: *TimeSlice*. The class *strdfs: SemiLinearPointSet* is the set of rational numbers that represent time values, longitude values, latitude values, altitude values, etc.

For example, there is a large sonic receiver that calls corresponding programs to analyze sound waves after receiving them. The software class has the instances Program1 and Program2, whose types are JavaProgram and PythonProgram, respectively. The receiver class has the instances Receiver1 and Receiver2, and Receiver2's type is SoundWaveReceiver. We compare the stRDF model and the stRDFS model by describing the spatiotemporal information. The stRDF model is as follows:

ex: program1	rdf: type	ex: JavaProgram
ex: program2	rdf: type	ex: PythonProgram
ex: program1	om: procedure	ex: CountProgram
ex: program2	om: procedure	ex: OutputProgram

ex: program1	om: hasPro1Call	ex: receiver2
ex: program2	om: hasPro2Call	ex: receiver2
ex: receiver2	rdf: type	ex: SoundWaveReceiver
ex: receiver2	ssn: measures	ex: sound
ex: receiver2	ssn: hasLocation	ex: location1
ex: location1	strdf: hasTrajectory	"(t = 8t and t = 18t or 9t ≤ t ≤ 14t) and k = 0t and ((20.9°N < L < 21°N and 45.8°E < D <= 46°E) or (L = 21°N and D = 46°E))" ^^ strdf: SemiLinearPointSet.

In the stRDF model, strdf: hasTrajectory describes the spatiotemporal data, which is considered to be the attribute value of strdf: hasTrajectory for Location1. In the stRDF model, only spatiotemporal data on whole entities can be recorded in the spatiotemporal dimension, while the spatiotemporal data on a certain part of the object cannot. For example, the stRDF model cannot represent the spatial data at t = 8t and record the temporal data when the object is in position (L = 21°N and D = 46°E). At the same time, if an attribute value changes at a certain time or in a certain spatial position, stRDF will record inaccurate data. The stRDFS model solves this problem as follows:

ex: program1	rdf: type	ex: JavaProgram
ex: program2	rdf: type	ex: PythonProgram
ex: program1	om: procedure	ex: AnalysisProgram
ex: program2	om: procedure	ex: AnalysisProgram
ex: program1	om:hasPro1Call	ex: receiver2
ex: program2	om: hasPro2Call	ex: receiver2
ex: receiver2	rdf: type	ex: SoundWaveReceiver
ex: receiver2	ssn: measures	ex: sound
ex: receiver2	ssn: hasLocation1	ex: location1
ex: receiver2	ssn: hasLocation2	ex: location2
om: hasPro1Call	strdfs: SpatiotemporalGeo	"t = 8 t and t = 18t and k = 0t and (20.9°N < L < 21°N and 45.8°E < D <= 46°E)" ^^ strdfs: SemiLinearPointSet
om: hasPro2Call	strdfs: SpatiotemporalGeo	"9t ≤ t ≤ 14t and L = 21°N and D = 46°E and k = 0t" ^^ strdfs: SemiLinearPointSet
ssn: hasLocation1	strdfs: SpatialGeometry	"(20.9°N < L < 21°N and 45.8°E < D <= 46°E)" ^^ strdfs: SemiLinearPointSet
ssn: hasLocation2	strdfs: SpatialGeometry	"(L = 21°N and D = 46°E)" ^^ strdfs: SemiLinearPointSet

Table 1. Topological relations among spatiotemporal classes.

Relation Name	Relation URI	Domain/Range
Equals	strdfs: geoEquals	strdfs: SpatialObject
Disjoint	strdfs: geoDisjoint	strdfs: SpatialObject
Meet	strdfs: geoMeet	strdfs: SpatialObject
Overlap	strdfs: geoOverlap	strdfs: SpatialObject
Covers	strdfs: geoCovers	strdfs: SpatialObject
CoveredBy	strdfs: geoCoveredBy	strdfs: SpatialObject
Inside	strdfs: geoInside	strdfs: SpatialObject
Contains	strdfs: geoContains	strdfs: SpatialObject
Before	strdfs: timBefore	strdfs: TemporalObject

Now	strdfs: timNow	strdfs: TemporalObject
After	strdfs: timAfter	strdfs: TemporalObject

At Location1, when t = 8t and t = 18t, the receiver calls Program1 to analyze the data. At Location2, when t ∈ [9t, 14t], the receiver calls Program2 to analyze the data. In the stRDFS model, spatiotemporal data are used to describe om: hasPro1Call and om: hasPro2Call, which facilitates the modification of common attribute values, such as program names. Based on this feature, the stRDFS model can represent spatiotemporal data and record changes in spatiotemporal attributes at any time or in any location.

We define 11 kinds of topological relations to describe the relations among spatiotemporal entities: *Equal*, *Disjoint*, *Meet*, *Overlap*, *Cover*, *CoveredBy*, *Inside*, *Contain*, *Before*, *Now*, and *After*. The corresponding domains are shown in Table 1.

3.3. Spatiotemporal RDF Graph Algebra

This subsection introduces five types of stRDFS graph algebras: union, intersection, difference, Cartesian product, and filter. In order to make these operations suitable for the model proposed in this paper, we improve the union, intersection, difference, and Cartesian product operations to provide solutions when the relationship between two points does not necessarily exist and add a filter operation to meet the specified requirements. These stRDFS graph algebras are sufficient and can satisfy the known operations. We provide an example in Figure 3 to show the algebraic process for stRDFS graphs by the method proposed in this paper.

Definition 5. Given two stRDFS graphs A (V_A, E_A, F_A, T_A, L_A) and B (V_B, E_B, F_B, T_B, L_B), the union of A and B is defined as $A \cup B = (V, E, F, T, L)$, where:

- $V = V_A \cup V_B$
- $E = E_A \cup E_B$
- $F = F_A \cup F_B$
- $\Pi_N T = \Pi_N T_A \cup \Pi_N T_B$
- $\Pi_k T = min(\Pi_N T_A, \Pi_N T_B)$
- $L = L_A \cup L_B$.

In Definition 5, $\Pi_{N/k}(x)$ represents the projection of x on N/k, where $N = [t_{s(f)}, t_{e(f)}]$ is the valid time and $k = t_{r(f)}$ is the reference time recorded at present. In $N = [t_{s(f)}, t_{e(f)}]$, $t_{s(f)}$ represents the start time of mapping $f_{vi\text{-}vj}$, and $t_{e(f)}$ represents the terminal time of mapping $f_{vi\text{-}vj}$. *min* represents the minimum reference time. The stRDFS graph requires that the reference time in the temporal data for each edge and each node must be consistent. When the union operation is performed over two stRDFS graphs, the reference time of the two graphs must be consistent. For convenience of description, we uniformly selected the minimum reference time as the reference time of the resulting graph.

As shown in Figure 3, there are two stRDFS graphs (a) and (b), and we performed the union operation in Definition 5 to obtain the graph (c). The solid line indicates that the relationship between two points must exist, and the dotted line indicates that the relationship between the two points does not necessarily exist.

Definition 6. Given two stRDFS graphs A (V_A, E_A, F_A, T_A, L_A) and B (V_B, E_B, F_B, T_B, L_B), the intersection of A and B is defined as $A \cap B = (V, E, F, T, L)$, where:

- $V = V_A \cap V_B$
- $E = E_A \cap E_B$
- $F = F_A \cap F_B$
- $\Pi_N T = \Pi_N T_A \cap \Pi_N T_B$
- $\Pi_k T = min\ (\Pi_N T_A, \Pi_N T_B)$
- $L = L_A \cap L_B$.

As shown in Figure 3, there are two stRDFS graphs (c) and (d), and we performed the intersection operation in Definition 6 to obtain the graph (e).

Definition 7. Given two stRDFS graphs A (V_A, E_A, F_A, T_A, L_A) and B (V_B, E_B, F_B, T_B, L_B), the difference of A and B is defined as $A - B = (V, E, F, T, L)$, where:

- $E = E_A - E_B$
- *V is the set of vertexes of E.*
- *F is the set of mappings of E.*
- *T is the temporal dataset of F.*
- *L is the spatial dataset of F.*

However, the algebraic operations of the abovementioned RDF graph are mainly used for the traditional RDF dataset, which cannot handle a complex spatiotemporal RDF dataset. Therefore, this paper aims to extend the traditional RDF graph operation and establish an RDF graph algebra for spatiotemporal data.

As shown in Figure 3, we can obtain graph (f) using Definition 7 with the difference of the two stRDFS graphs (d) and (a).

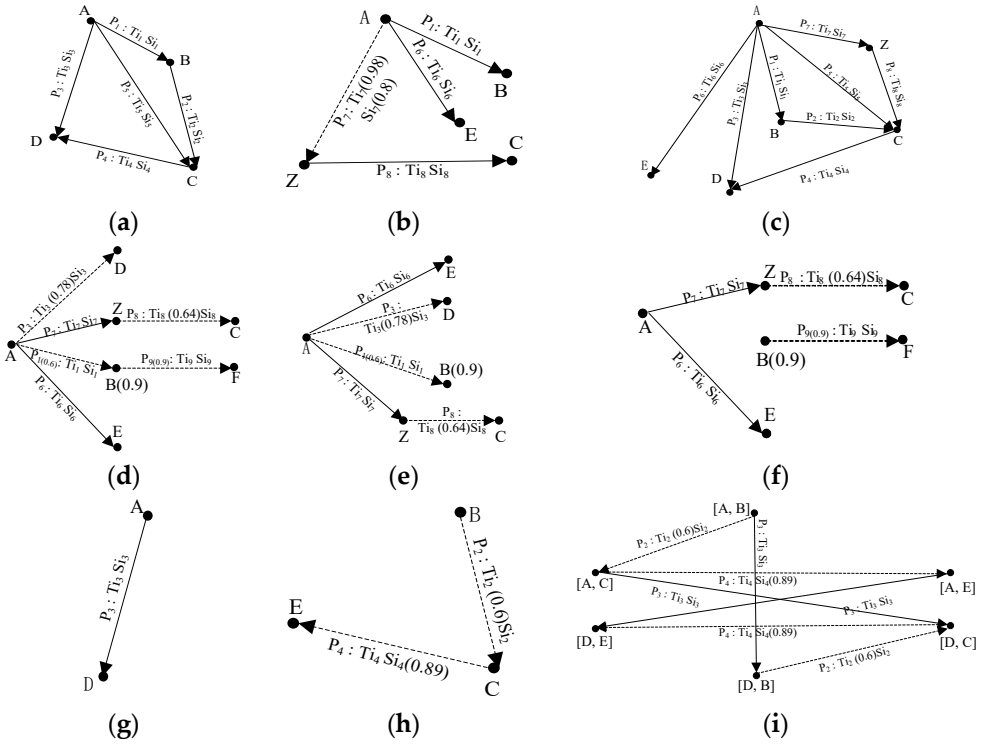


Figure 3. Algebraic operations of stRDFS graphs.

Definition 8. Given two stRDFS graphs A (V_A, E_A, F_A, T_A, L_A) and B (V_B, E_B, F_B, T_B, L_B), the Cartesian product of A and B is defined as $A \times B = (V, E, F, T, L)$, where:

- $V = V_A \times V_B$
- $E = \{(u, u_2)(u, v_2) \mid u \in V_A, u_2 v_2 \in E_B\} \cup \{(u_1, \omega)(v_1, \omega) \mid \omega \in V_B, u_1 v_1 \in E_A\}$
- *F is the set of mappings of E.*
- *T is the temporal dataset of F.*
- *L is the spatial dataset of F.*

There are two stRDFS graphs (g) and (h) in Figure 3. We can obtain graph (i) with their Cartesian product using Definition 8.

Theorem 3. *Given two stRDFS graphs A (V_A, E_A, F_A, T_A, L_A) and B (V_B, E_B, F_B, T_B, L_B), we have the following.*

- *$A \cup B$ is an stRDFS graph.*
- *$A \cap B$ is an stRDFS graph.*
- *$A - B$ is an stRDFS graph.*
- *$A \times B$ is an stRDFS graph.*

Definition 9 (stRDFS graph pattern). An stRDFS graph pattern is defined as $P = (V_P, E_P, F_P, T_P, L_P, Re)$, where:

- *V_P is a finite set of vertexes.*
- *E_P is a finite set of directed edges.*
- *F_P is the set of mappings of E_P.*

- T_P *is a temporal dataset.*
- L_P *is a spatial dataset.*
- $R_e = \{R_1R_2, R_1|R_2, R+, \varepsilon\}$ *is a set of filter rules describing* E_p*, where R represents a filter rule.*

In practice, the stRDFS graph needs to be filtered according to certain rules so that the resulting graph meets the specified requirements. The stRDFS graph pattern described in Definition 9 represents the filter rules. In the program, the input is often filter rules and the stRDFS graph that needs to be processed, and the output is the resulting graph that satisfies specified requirements. According to the stRDFS graph filter rules, we can obtain the following conclusions: V_p is a vertex pattern abstracted from the rules and each vertex pattern corresponds to a set of vertexes in the filtered stRDFS graph. E_p is an abstraction of the attributes in the filter rules and each edge represents a type of edge of the stRDFS graph being filtered. F_p is an abstract collection of attribute names in the rule. T_p is the temporal dataset in the filter rules. Similarly, L_p is the spatial dataset in the filter rules. *Re* is regular expression and can have four forms: R_1R_2, $R_1|R_2$, R^+, and ε. R_1R_2 is a concatenation of expressions that represents that two rules are next to each other. $R_1|R_2$ is an alternative of expressions that represents that filtering can be done by meeting one of the two rules. R^+ denotes one or more occurrences of R. ε represents that there are no filter rules to filter stRDF graphs.

For example, consider an stRDFS graph pattern P, which models information concerning a writer (? w) who is born in A city, the fee for a book (? book) that the writer had written from 2016 to 2018 in A city, which is more than \$10 (? f > \$ 10), and the book's genre, which is comedy. P can be expressed in the form of Figure 4.

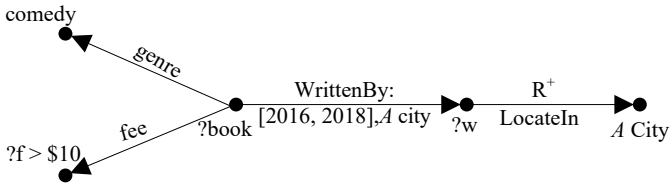


Figure 4. The stRDFS graph pattern *P*.

Using the stRDFS graph G shown in Figure 5, we performed the filter operation in Definition 9. The result of spatiotemporal filtering of P and G is shown in Figure 6.

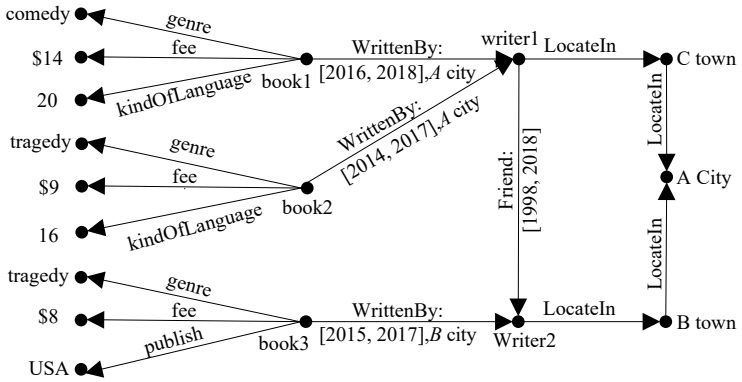


Figure 5. The stRDFS graph pattern *G*.

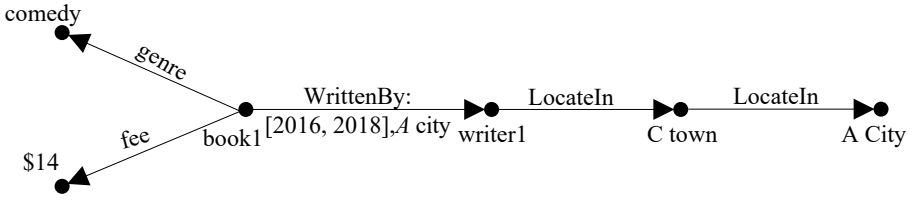


Figure 6. The result of performing the filter operation over *P* and *G*.

4. Spatiotemporal RDF Syntax Specification

In the stRDFS model, we define several topological relations to describe the relations among spatiotemporal entities and introduce five types of stRDFS graph algebras. However, stSPASQL does not define corresponding functions to deal with spatiotemporal data queries. While the queries contain the operations we proposed, stSPASQL cannot meet the requirements of the queries and needs to be improved. To solve this problem, we specify a spatiotemporal RDF syntax on the basis of stSPASQL [11]. The syntax proposed in this subsection is similar to the SQL syntax, and it is designed so that stRDFS format datasets can be easily accessed. It also provides a number of advanced functions for building more expressive queries that illustrate other filter conditions and then format the final output. The overall structure of the syntax is similar to that of SQL, which has three main parts represented by the uppercase keywords SELECT, FROM, and WHERE.

- The keyword SELECT determines the specification of the query result. It is followed by the rest of the query: the list of identifiers of the query result.
- The keyword FROM specifies the query's scope, which is normally a spatiotemporal dataset or several stRDFS graphs.
- The keyword WHERE represents the actual query and is followed by the query criteria, which are given by a pattern that corresponds to some stRDFS graphs replaced by variables. More complicated patterns are also allowed to be formed with some algebraic operators. This mode can be used not only to query a single spatiotemporal dataset but also multiple spatiotemporal datasets.

Table 2. Spatiotemporal functions.

Function	Type	Syntax format	Meaning
strdfs: distance	Spatial Function	FILTER (strdfs: distance(A, B))	Returns the shortest distance between A and B
strdfs: intersection		FILTER (strdfs: intersection (A, B))	Returns a geometric object that represents all points in the intersection of A and B
strdfs: union		FILTER (strdfs: union (A, B))	Returns a geometric object that represents all points in the union of A and B
strdfs: difference		FILTER (strdfs: difference (A, B))	Returns a geometric object that represents all points in the difference set of A and B
strdfs: spCartesian		FILTER (strdfs: spCartesian (A, B))	Returns a geometric object that represents all points in the Cartesian product set of A and B
strdfs: spfilter		FILTER (strdfs: spfilter (A, B))	Returns a geometric object that represents all points in the subgraph of A after filtering according to the pattern B
strdfs: spEquals		FILTER (strdfs: spEquals (?A, ?B))	The query result satisfies A = B
strdfs: spDisjoint		FILTER (strdfs: spDisjoint (?A, ?B))	The query result satisfies that A is disjoint with B
strdfs: spMeet		FILTER (strdfs: spMeet (?A, ?B))	The query result satisfies that A meets B
strdfs: spOverlap		FILTER (strdfs: spOverlap (?A, ?B))	The query result satisfies that A overlaps B
strdfs: spCovers		FILTER (strdfs: spCovers (?A, ?B))	The query result satisfies that A covers B
strdfs: spCoveredBy		FILTER (strdfs: spCoveredBy (?A, ?B))	The query result satisfies that A is covered by B
strdfs: spInside		FILTER (strdfs: spInside (?A, ?B))	The query result satisfies that A is inside B
strdfs: spContains		FILTER (strdfs: spContains (?A, ?B))	The query result satisfies that A contains B
strdfs:spEnvelope		FILTER (strdfs: spEnvelope (A))	Returns the minimum bounding box of A

strdfs: spBefore	Temporal Function	FILTER (strdfs: spBefore (?A, ?B))	The query result satisfies that A is earlier than B
strdfs: spNow		FILTER (strdfs: spNow (?A, ?B))	The query result satisfies that A and B are at the same time
strdfs: spAfter		FILTER (strdfs: spAfter (?A, ?B))	The query result satisfies that A is later than B

The function indicates the relations between the variables expressed by the FILTER statement, which is a part of the WHERE statement. In order to browse the spatiotemporal syntax, we define the spatial functions and temporal functions shown in Table 2.

In order to demonstrate the application of the proposed functions, we provide some examples. Examples 1, 2, and 3 represent queries of spatial data, temporal data, and spatiotemporal data, respectively.

Example 1. When we query the name and spatial data of a destroyed forest less than 0.1 km from a city, the syntax is as follows:

```
SELECT ?NAME ?FGEO
WHERE {
?R rdf: type noa: Region
?R strdfs: SpatialGeometry ?RGEO
?R noa: hasCorineLandCoverUse ?S
?S rdfs: subclassof clc: Forests
?C rdf: type dbpedia: City
?C strdfs: SpatialGeometry ?CGEO
?NAME rdf: type noa: DestroyedArea
?NAME strdfs: SpatialGeometry ?FGEO
FILTER(strdfs: spInside (?RGEO, ?FGEO) && strdfs: distance (?FGEO, ?CGEO) < 0.1))
}
```

The SELECT statement is followed by a list of the names of the results, namely: the region name and the spatial data on the region. The WHERE statement states the relations of the triples that need to be satisfied between the following variables: ?R indicates the area containing the forest ?F, ?C indicates the mentioned city, and ?NAME represents the mentioned destroyed forest. The FILTER statement represents the spatial function of spatial variables: strdfs: spInside (?RGEO, ?FGEO) represents that ?R is inside ?F and strdfs: distance (?FGEO, ?CGEO) < 0.1 represents that the shortest distance between ?F and ?C is less than 0.1 kilometers.

Example 2. When we query the temporal data in a program that is called simultaneously with a certain other program, the syntax is as follows:

```
SELECT ?P ?PTS
WHERE {
?P rdf: type ex: Program
?P om: hasProCallP ex: TimeSlice1
om: hasProCallP strdfs: TimeSlice ?PTS
?Q rdf:type ex: Program
?Q om: hasProCallQ ex: TimeSlice2
om: hasProCallQ strdfs: TimeSlice ?QTS
FILTER (strdfs: spNow (?PTS, ?QTS))
}
```

Since the program's name and the temporal data of the program are sought, the SELECT statement is followed by ?P and ?PTS, that is, the temporal information of ?P is ?PTS. FILTER (strdfs: spNow (?PTS, ?QTS)) represents that ?P and ?Q have the same temporal information.

Example 3. On the basis of the example in Section 3.2, when we query the name of the programs that are called by the sound wave receiver in the adjacent area and its spatiotemporal data, the syntax is as follows:

```
SELECT ?P ?PGT
```

```
WHERE {
?P rdf: type ex: Program
?P om: hasCallPBy ex: SoundWaveReceiver
om: hasCallPBy strdfs: SpatiotemporalGeo
?PGT (?PGEO, ?PTS)
?Q rdf:type ex: Program
?Q om: hasCallQBy ex: SoundWaveReceiver
om: hasCallQBy strdfs: SpatiotemporalGeo
?QGT (?QGEO, ?QTS)
FILTER (strdfs: spMeet (?PGEO, ?QGEO) && strdfs:spNow (?PTS, ?QTS))
}
```

Since the program's name and the spatiotemporal data of the program are sought, the SELECT statement is followed by ?P and ?PTS, that is, the spatiotemporal information of ?P is ?PGT. FILTER (strdfs: spMeet (?PGEO, ?QGEO) && strdfs: spNow (?PTS, ?QTS)} means that ?P meets ?Q and they have the same temporal information.

This paper has a proposed spatiotemporal data model and the corresponding algebraic operations. A lot of missing or insufficient operations that do not exist in conventional models have also been put forward. Little related work on spatiotemporal RDF algebras has involved experimental results, so we consider only the theoretical feasibility of the proposed method in this paper. However, there are several intermediate algebras for optimizing RDF graph pattern matching, and they were proposed from a matching or querying viewpoint, not the algebra viewpoint. As a result, we will continue to study the corresponding querying method in our future work.

5. Conclusion and Future Work

In this paper, we explored a spatiotemporal RDF semantics and defined a spatiotemporal data model based on the RDF. Based on this model, we proposed algebraic operations for manipulating spatiotemporal RDF data. In our method, in contrast to previous works, filtering rules are contained in the algebraic operations so that the resulting graph meets the specified requirements. This algebra consists of a series of operations that make it possible to express the content of data and the structure of a spatiotemporal RDF graph. Furthermore, we also introduced a syntax specification for the spatiotemporal RDF, which may contribute to the study of spatiotemporal RDF querying in the future.

There are three aspects left for future work. The first is that it is worth trying to consider the application of the query syntax and improve efficiency without affecting accuracy. The second is that, in the face of uncertainty in spatiotemporal data, a more flexible model should be built. The last is that the query language of the spatiotemporal RDF should be explored and compared with other state-of-the-art methods.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (Grant No. 61402087), the Natural Science Foundation of Hebei Province (Grant No. F2019501030), the Natural Science Foundation of Liaoning Province (Grant No. 2019-MS-130), and the Fundamental Research Funds for the Central Universities (Grant No. N172304026).

Author Contributions: Luyi Bai conceived the idea and provided continuous advise during the design phase. Lin Zhu and Nan Li designed the method and wrote the paper. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; and in the decision to publish the results.

References

1. Lassila, O. Resource description framework (RDF) model and syntax specification, W3C recommendation. Available online: http://www. w3. org/TR/PR-rdf-syntax. (accessed on 22, February, 1999).

2. Batsakis, S.; Petrakis, E.G.M. Representing temporal knowledge in the semantic web: The extended 4d fluents approach. In *Combinations of intelligent methods and applications*. Springer: Berlin, Heidelberg, 2011; pp. 55-69.
3. Lutz, C.; Wolter F.; Zakharyaschev, M. Temporal description logics: A survey. In Proceedings of 15th International Symposium on Temporal Representation and Reasoning, Montreal, QC, Canada, 16-18, June, 2008; IEEE; pp. 3–14.
4. Reed, T.W.; McMeekin, D.A.; Reitsma, F. Representing spatial relationships within smart cities using ontologies. In *Information Innovation Technology in Smart Cities*; Springer: Berlin, Germany, 2018; pp. 33–45.
5. Kim, J.J.; Shin, I.S.; Lee, Y.S., et al. Spatio-Temporal Ontology Management Systems for Semantic Web. Information, 2016, 19(9B), pp. 4237–4254.
6. Tappolet, J.; Bernstein, A. Applied temporal RDF: Efficient temporal querying of RDF data with SPARQL. In Proceedings of European Semantic Web Conference, Crete, Greece, 31 May-4 June, 2009; Springer: Berlin, Heidelberg, pp. 308–322.
7. Hernández, D.; Hogan, A.; Krötzsch, M. Reifying RDF: What Works Well With Wikidata? In Proceedings of the 11th International Workshop on Scalable Semantic Web Knowledge Base Systems, Bethlehem, Pennsylvania, USA, 11, October, 2015.
8. Gutierrez, C.; Hurtado, C.; Vaisman, A. Temporal rdf. In Proceedings of European Semantic Web Conference, Crete, Greece, 29 May-1 June, 2005; Springer: Berlin, Heidelberg, pp.93–107.
9. Pugliese, A.; Udrea, O.; Subrahmanian, V.S. Scaling RDF with time. In Proceedings of the 17th International Conference on World Wide Web, Beijing, China, 21-25, April, 2008, pp. 605–614.
10. Consoli, S.; Mongiovic, M.; Nuzzolese, A.G., et al. A smart city data model based on semantics best practice and principles. In Proceedings of the 24th International Conference on World Wide Web, Italy, 18-22, May, 2015, pp. 1395–1400.
11. Koubarakis, M.; Kyzirakos, K. Modeling and querying metadata in the semantic sensor web: The model stRDF and the query language stSPARQL. In Proceedings of Extended Semantic Web Conference. Crete, Greece, 30 May-3, June, 2010, Springer, Berlin, Heidelberg, pp. 425–439.
12. Sheng, G.L.; Su, Y.L.; Wang, W.D. A new fractal approach for describing induced-fracture porosity/permeability/compressibility in stimulated unconventional reservoirs. *J. Petroleum Sci. Eng.* **2019**, *179*, 855–866.
13. Zhao, H.; Xu, L.; Guo, Z., et al. A new and fast waterflooding optimization workflow based on INSIM-derived injection efficiency with a field application. *J. Petroleum Sci. Eng.* **2019**, *179*, 1186–1200.
14. Huang, L.; Hu, Y.; Li, Y., et al. A study of regular and irregular neutrosophic graphs with real life applications. *Mathematics* **2019**, *7*, 551.
15. Nikitopoulos, P.; Vlachou, A.; Doulkeridis, C.; Vouros, G.A. Parallel and scalable processing of spatio-temporal RDF queries using Spark. *GeoInformatica*, 2019, DOI: https://doi.org/10.1007/s10707-019-00371-0.
16. Frasincar, F.; Houben, G.J.; Vdovjak, R., et al. RAL: An algebra for querying RDF. World Wide Web, 2004, 7(1), pp. 83-109.
17. Robertson, E.L. Triadic relations: An algebra for the semantic web. In Proceedings of International Workshop on Semantic Web and Databases, Toronto,Canada, 2004.
18. Chen, L.; Gupta, A.; Kurul, M.E. A semantic-aware RDF query algebra. In Proceedings of the International Conference on Management of Data (COMAD), Hyderabad, India, 20–22, December, 2005.
19. Jamour, F.; Abdelaziz, I.; Kalnisk P. A demonstration of magiq: matrix algebra approach for solving rdf graph queries. In Proceedings of the VLDB Endowment, 2018, 11(12), pp. 1978-1981.
20. Thakkar, H.; Punjani, D.; Auer, S., et al. Towards an integrated graph algebra for graph pattern matching with Gremlin. In Proceedings of International Conference on Database and Expert Systems Applications, Lyon, France, 2017; Springer, Cham, pp. 81-91.
21. Córcoles, J.E.; González, P. A specification of a spatial query lan-guage over GML. In Proceedings of the 9th ACM internationalsymposium on advances in geographic information systems, Atlanta, Georgia, USA, 2001.
22. Pan, F.; Hobbs, J.R. Temporal Aggregates in OWL-Time. In Proceedings of FLAIRS conference, 2005, 5, pp. 560-565.
23. Baratis, E.; Petrakis, E.G.M.; Batsakis, S.; Maris, N.; Papadakis, N. TOQL: Temporal ontology querying language. In Proceedings of International Symposium on Spatial and Temporal Databases, Aalborg, Denmark, 2009; Springer: Berlin, Heidelberg, pp. 338-354.

24. Batsakis, S.; Petrakis, E.G.M. Representing temporal knowledge in the semantic web: The extended 4d fluents approach. In *Combinations of intelligent methods and applications*. Springer: Berlin, Heidelberg, 2011; pp. 55-69.
25. Moffitt, V.Z.; Stoyanovich, J. Temporal graph algebra. In Proceedings of the 16th International Symposium on Database Programming Languages. Munich, Germany, 2017; ACM, pp. 10.
26. Perry, M.; Sheth, A.; Arpinar, I.B.; Hakimpour, F. Geospatial and temporal semantic analytics. In *Handbook of research on geoinformatics*. IGI Global, 2009; pp. 161-170.
27. Hakimpour, F.; Aleman-Meza, B.; Perry, M.; Sheth, A. Spatiotemporal-thematic data processing for the semantic web. In *The Geospatial Web*. Springer: London, 2009; pp. 79-89.
28. Perry, M.; Jain, P.; Sheth, A.P. SPARQL-ST: Extending SPARQL to Support Spatiotemporal Queries. Geospatial Semantics and the Semantic Web, 2011, 12, pp. 61-86.
29. Bai, L.; Xu, C. Spatiotemporal query algebra based on native XML. In *Handbook of research on innovative database query processing techniques*. IGI Global: Hershey, Pennsylvania, USA, 2015; pp. 275-293.
30. Bai, L.; Zhu, L. An Algebra for Fuzzy Spatiotemporal Data in XML. IEEE Access, 2019, 7, pp. 22914-22926.
31. Cyganiak, R. A relational algebra for SPARQL. Digital Media Systems Laboratory HP Laboratories Bristol. HPL-2005-170, 2005, 35, pp. 9.
32. Perry, M.; Herring, J. OGC GeoSPARQL-A geographic query language for RDF data. OGC implementation standard, 2012, 40.
33. Battle, R.; Kolas, D. Geosparql: enabling a geospatial semantic web. Semantic Web Journal, 2011, 3(4), pp. 355-370.
34. Battle, R.; Kolas, D. Enabling the geospatial semantic web with parliament and geosparql. Semantic Web, 2012, 3(4), pp. 355-370.
35. Zuo, C.; Pal, A.; Dey, A. New Concepts of Picture Fuzzy Graphs with Application. Mathematics, 2019, 7(5), pp. 470.
36. Ma, Z.; Li, G.; Yan, L. Fuzzy data modeling and algebraic operations in RDF. Fuzzy Sets and Systems, 2018, 351, pp. 41-63.
37. Dey, A.; Pal, A.; Long, H.V. Fuzzy minimum spanning tree with interval type 2 fuzzy arc length: formulation and a new genetic algorithm. Soft Computing, 2019, pp. 1-12.
38. Nitta, K.; Savnik, I. A Distributed Query Execution Method for RDF Storage Managers. In Proceedings of the 10th International Workshop on Scalable Semantic Web Knowledge Base Systems Co-Located with 13th International Semantic Web Conference, Riva del Garda, Trentino, Italy, 2014.
39. Ravindra, P.; Kim, H.S.; Anyanwu, K. An intermediate algebra for optimizing RDF graph pattern matching on MapReduce. In Proceedings of Extended Semantic Web Conference, Berlin, Germany, 2011; Springer, Berlin, Heidelberg, pp. 46-61.
40. Wang, D.; Zou, L.; Zhao, D. gst-Store: An Engine for Large RDF Graph Integrating Spatiotemporal Information. Institute of Computer Science & Technology, 2014.
41. Lee, K.S.; Lee, K.Y.; Kim, Y.H.; Choi, J.J.; Jang, G.S. Spatio-Temporal Ontology for the Semantic Web. International Information Institute (Tokyo) Information, 2015, 18(1), pp. 329-334.
42. Perry, M.; Sheth, A.P.; Hakimpour, F.; Jain, P. Supporting complex thematic, spatial and temporal queries over semantic web data. In Proceedings of 2nd International Conference on Geospatial Semantics, Mexico City, Mexico, 2007; pp. 228–246,.
43. Perez, J.; Arenas, M.; Gutierrez, C. Semantics and complexity of SPARQL. In Proceedings of 5th International Semantic Web Conference, Athens, GA, USA, 2006; pp. 30–43.
44. Perry, M.; Estrada, A.; Das, S.; Banerjee, J. Developing GeoSPARQL Applications with Oracle Spatial and Graph. In proceedings of ISWC, 2015; pp. 57-61.
45. Wang, D.; Zou, L.; Zhao, D. gst-store: Querying Large Spatiotemporal RDF Graphs. Data and Information Management, 2017, 1(2), pp. 84-103.