An Efficient Algorithm of Star Subgraph Queries on Urban Traffic Knowledge Graph

Tao Sun · Jianqiu Xu · Caiping Hu

Received: 19 July 2022 / Revised: 9 September 2022 / Accepted: 6 October 2022 / Published online: 21 October 2022
© The Author(s) 2022

Abstract
Knowledge graph has wide applications in the field of computer science. In the knowledge service environment, the information is large and explosive, and it is difficult to find knowledge of common phenomena. The urban traffic knowledge graph is a knowledge system that formally describes urban traffic concepts, entities and their interrelationships. It has great application potential in application scenarios such as user travel, route planning, and urban planning. This paper first defines the urban traffic knowledge graph and the star subgraph query of the urban traffic knowledge graph. Then, the road network data and trajectory data are collected to extract the urban traffic knowledge, and the urban traffic knowledge graph is constructed with this knowledge. Finally, a star subgraph query algorithm on the urban traffic knowledge graph is proposed. The discussion of the star subgraph query mode gives the corresponding application scenarios of our method in the urban traffic knowledge graph. Experimental results verify the performance advantages of this method.

Keywords Knowledge graph · Subgraph matching · Urban traffic data · Property graph

1 Introduction

Over the past 30 years, modeling concepts and structures have been continuously explored for rational integration of time in GIS. Most of these models are aimed at integration, management and analysis of space, time-moving data and phenomena, and have wide applications in the environment, the city and the geographical domain [1]. Although a series of conceptual and formal advancements have been made on various fronts, new mechanisms for modeling spatiotemporal knowledge are still needed, providing additional data representation and analysis capabilities for moving objects and complex spatiotemporal dynamics. Continuous advancements in real time and sensor technology facilitate the integration of large geographic datasets. However, this often leads to different data structures and complex difficult operations. But with the development of artificial intelligence technology, we have a new way: knowledge graph.

In recent years, due to the advancement of intelligent system applications, knowledge graphs play an important role in more and more application scenarios, providing diverse knowledge services for various intelligent tasks. Among them, there are a large number of application scenarios whose systems are highly sensitive to changes in knowledge over time, such as application scenarios related to moving object trajectories. This makes the traditional static knowledge graph unable to meet the needs of dynamic knowledge changes. In this regard, urban traffic knowledge graphs containing temporal information and spatial information can be a feasible solution and become a current research hotspot. The urban traffic knowledge graph can completely record the process of knowledge change, ensure the immediacy and validity of knowledge, and enable subsequent tasks to obtain better knowledge services of temporal awareness, such that it has high practical application value.

Figure 1 shows a subgraph snapshot of the urban traffic knowledge graph. The blue node represents the entity in the urban traffic knowledge graph. The green node represents the attributes of the entity. The directed edges between nodes express the relationship between them. Among them, there are two moving objects of taxi drivers Zhang and Li, a road called Yixing Road and a fast food restaurant McDonald’s. Taxi drivers have name attributes and speed attributes, and
have relationships with road entities such as start, pass and stop. McDonald’s is a point of interest (POI), which has two attributes, namely the category attribute and the name attribute. There is a relationship called locateat, which indicates that the POI is located at the road entity. Yixing Road is a road entity with attributes such as name and type and has various topological relationships with other entities such as roads, road segments, and moving objects. Tables 1, 2, and 3 show the entities, relationships, and attributes that exist in the subgraph snapshots of the urban traffic knowledge graph, respectively. In addition, Fig. 1 is not only an example of an urban traffic knowledge graph but also an example of a star subgraph query. Figure 1 is a star subgraph with four central nodes and eight adjacent nodes.

This paper mainly considers the application of star subgraph query in urban traffic knowledge graph. In order to realize the application of the star subgraph query on the urban traffic knowledge graph, we construct the urban traffic knowledge graph. A star graph is a graph consisting of star center nodes and their adjacent nodes. In the star graph, the star center node can fully describe the characteristic information of the adjacent nodes. In the urban traffic knowledge graph, there are a lot of applications of
We explore and study the query pattern of star subgraphs. For example, the subgraph of the road network layer combined with the trajectory information can reflect the knowledge of traffic flow, which can be used for route planning. A person and his friends who meet in a certain place can form a star subgraph of the trajectory layer to provide knowledgeable assistance for scenarios such as epidemic prevention and control. If a person frequently visits fast food restaurants and residential areas, then we can construct his trajectory as a star subgraph to predict the person’s occupation.

At present, the related research on the urban traffic knowledge graph mainly includes the discovery, representation, and reasoning of urban traffic knowledge. However, due to the need for different types of multi-directional data, there are relatively few related works on the construction scheme and query method of urban traffic knowledge graph at this stage. There are several subgraph query algorithms, but they mainly focus on the topology of the graph. In order to construct and query the urban traffic knowledge graph, it is necessary to consider not only the topology of the urban traffic knowledge graph but also the temporal and spatial characteristics of the urban traffic knowledge graph itself. Therefore, the research on the construction and query method of the urban traffic knowledge graph is of great significance in strengthening and improving the related research fields of the urban traffic knowledge graph. We first define the urban traffic knowledge graph and the star subgraph query of the urban traffic knowledge graph. Then we collect road network data and trajectory data, and extract urban traffic knowledge from them. Based on this knowledge, a knowledge graph of urban traffic is constructed. Finally, a star subgraph query algorithm on the urban traffic knowledge graph is proposed, and the discussion of the star subgraph query mode gives the corresponding application scenarios of our method in the urban traffic knowledge graph. The experimental results show the advantages of this method. The main contributions of this paper are as follows:

- We collect data containing temporal and spatial features such as road network data and trajectory data. Then we extract the knowledge needed in the urban traffic knowledge graph from these data containing temporal and spatial features, thereby constructing the urban traffic knowledge graph.
- We explore and study the query pattern of star subgraphs on the urban traffic knowledge graph. Based on this, an efficient star subgraph query algorithm TKG_Sub based on urban traffic knowledge graph is proposed. The algorithm uses a filtering mechanism to exclude queries whose time range is beyond the data graph, improving the query efficiency on the urban traffic knowledge graph with temporal features.
- We run TKG_Sub and other three subgraph query algorithms on the constructed urban traffic knowledge graph to test and compare the query efficiency of TKG_Sub. The experimental results show that our method has less query response time than other algorithms, and is at least 20% faster than other methods in the case of 27 nodes. Therefore our method is effective.

The rest of this paper is organized as follows. Section 2 introduces the relevant work of knowledge graph and subgraph searching. In Sect. 3 we propose the definition and concept of the urban traffic knowledge graph. In Sect. 4, we extract entities and relations to construct the urban traffic knowledge graph. In Sect. 5, we propose a star subgraph matching algorithm TKG_Sub on the urban traffic knowledge graph and discuss related application scenarios. In Sect. 6, the performance of the algorithm is shown by the experimental program and the result analysis. Finally, Sect. 7 concludes the paper.

2 Related Works

The early concept of knowledge graph originated from the idea of the Semantic Web by Tim Berners-Lee, the father of the World Wide Web. It hopes to use the graph structure to model and record the relationship and knowledge between all things in the world, so as to achieve a more accurate and efficient object-level search. Knowledge graphs were originally proposed by Google in 2012 as a knowledge base used to enhance the functions of search engines [2]. Essentially, a knowledge graph is a semantic network that reveals the relationships between entities, can effectively represent data resources, can efficiently find complex related information, and has semantic processing capabilities.

In recent years, with the rapid development of a lot of fields such as natural language processing, deep learning, and graph data processing, knowledge graphs have made much new progress in the fields of automated knowledge acquisition, knowledge representation learning and reasoning, large-scale graph mining and analysis. The relevant techniques of knowledge graph are widely used in a number of fields such as search engines, intelligent question answering, language understanding, recommendation computing, and massive data decision analysis. Knowledge graph is one of the essential and important techniques to realize artificial intelligence at the cognitive level.

The development of the Internet has provided new opportunities for knowledge engineering. To a certain extent, the emergence of the Internet has helped traditional knowledge engineering to break through the bottleneck of knowledge acquisition. Most of the knowledge graphs built in this period are general knowledge graphs, which are oriented...
to all fields. The general knowledge graph is mainly used in Internet-oriented business scenarios such as search, recommendation, and question answering. Representative large-scale general knowledge graphs include Freebase [3], DBpedia [4], Wikidata [5] and YAGO [6]. Typical Chinese open general knowledge graphs include Zhishi.me [7], CN-DBpedia [8], and Xlore [9]. These Chinese knowledge graphs are mainly gathered in OpenKG, a community project for open knowledge graphs in the Chinese domain. In addition to general large-scale knowledge graphs, various industries are also building knowledge graphs in vertical domains. The domain knowledge graph is relative to the general knowledge graph, it is a knowledge graph for a specific field, such as the knowledge graph in the medical field [10], the knowledge graph in the maritime transportation field [11] and the knowledge graph in the public safety field [12].

One of the main research works based on spatiotemporal data is spatiotemporal knowledge reasoning, and the focus of spatiotemporal knowledge reasoning is mainly on the prediction of temporal connections of future relationships. Cunchao Zhu et al. proposed a temporal knowledge graph representation learning model named CyGNNet [13], which is used for fact prediction or relation prediction in the temporal knowledge graph. It is based on a new time-aware replication generation mechanism, and its central idea is future facts can be predicted based on historical facts. Rakshit Trivedi et al. proposed Know-Evolve, a deep evolutionary knowledge network architecture that learns from the historical evolution of entity embeddings in a specific relational space. The network architecture estimates whether it holds true at time t based on the state of a fact at time t-1 [14]. Woojeong Jin et al. proposed the Recurrent Event Network RE-NET, a novel autoregressive architecture for predicting future interactions, addressing spatiotemporal knowledge reasoning by simulating temporal, multi-relational, and concurrent interactions between entities [15]. Inspired by the regular differential equations (NODE), Zifeng Ding et al. extended the idea of the continuous depth model to the time-evolving multi-relational graph data and proposed a new NODE model for predicting the time knowledge graph of the regular differential equations. The model captures temporal information through neural regular differential equations and structural information through graph neural networks [16].

There are also a lot of scholars who have constructed knowledge graphs in the space-time domain and conducted research on their applications. Chunqing Xiao et al. proposed a temporal knowledge graph incremental construction model [17]. When interactions occur, the model automatically extracts semantic paths of different lengths between users and items. The model then uses the recursive neural network and the standard multilayer perceptron (MLP) to collect path semantic information and interaction itself information of different lengths and update the entity representation. Finally, the model uses MLP to predict the probability that users will like an item after seamlessly integrating these changes into a unified representation to improve recommendation performance. Chenyi Zhuang et al. analyzed GPS points from temporal, spatial, and spatiotemporal views, constructed an urban movement knowledge graph with embedded temporal and spatial information, and then applied the knowledge graph to predict users’ attention to different locations in the city [18]. In order to realize the intelligent service of emergency decision-making, Jiahui Chen et al. used the urban traffic knowledge graph as the information management framework [19]. Jiuyuan Tan et al. used a top-down method to construct the knowledge graph of the urban transportation system and adopted a knowledge inference model based on learning to realize knowledge completion, mining the implicit relationship between traffic entities and discovering traffic knowledge [20]. Huandong Wang et al. propose a new type of knowledge graph called Spatio-temporal urban knowledge graph (STKG) [21], where mobility trajectories, category information of venues, and temporal information are jointly modeled by the facts with different relation types in STKG. They use a complex embedding model with elaborately designed scoring functions to measure the plausibility of facts in STKG to solve the knowledge graph completion problem, which considers temporal dynamics of the mobility patterns and utilizes POI categories as the auxiliary information and background knowledge. Yin Hao Sun et al. mainly study how to speed up subgraph matching on knowledge graphs [22]. They propose a new subgraph matching algorithm based on the knowledge graph subgraph index, called FGqt-Match. The algorithm consists of two parts. The first part is a subgraph index of a matching-driven flow graph, which reduces redundant calculations in advance. The second part is a multi-label weight matching-driven flow graph, which evaluates a near-optimal matching tree for minimizing the intermediate candidates. Yuan Li et al. focus on the problem of cohesive subgraph search in large temporal graphs [23]. Given a query vertex, they find a continuous dense subgraph that includes the query vertex. Furthermore, they propose an approximate local search method called Approx-LS, which greedily expands the current subgraph guided by the developed heuristic functions until identifying the results.

An urban traffic knowledge graph is a knowledge graph that contains both time and spatial information. The knowledge contained in the time knowledge graph changes in time and space as compared with the usual static knowledge graph. The urban traffic knowledge graph can provide knowledge with temporal and spatial information for subsequent tasks. Related works around urban traffic knowledge graphs mainly include representation, reasoning, and question answering of urban traffic knowledge. However, there are few types of research on urban traffic knowledge graph
construction methods at this stage, and the challenges of the
growing scale of urban traffic knowledge graphs cannot be
ignored. Therefore, the research on the construction method
of urban traffic knowledge graphs is of great significance in
strengthening and improving the related research fields of
the urban traffic knowledge graph.

In order to study the urban traffic data, this paper con-
structs an urban traffic knowledge graph and uses the star
subgraph query method to retrieve the knowledge in the
urban traffic knowledge graph, which can discover the rela-
tionships between entities in the urban traffic knowledge
graph. For example, from which road segment the moving
object starts, which road segment passes through, and which
road segment finally reaches.

3 Urban Traffic Knowledge Graph

This section first introduces how knowledge is represented
by triples in common knowledge graphs, and how it is rep-
resented by quadruples under the condition that knowledge
has temporal features. Then we introduce the model of the
urban traffic knowledge graph and its components.

3.1 Preliminary

A knowledge graph is a graph-structured knowledge base
that contains human knowledge and facts. A traditional static
knowledge graph can be defined as a set of triples \((s, p, o)\),
where \(s\) represents the subject entity, \(o\) represents the object
time, and \(p\) represents the relationship between the subject
entity and the object entity. However, the relationship of two
specific entities may change over time, which means that the
truth expressed by the triple may not always be true. As an
example, consider the following facts:

\[(\text{Road1}, \text{intersect}, \text{Road2}),\]
\[(\text{Wang, pass, Road1}),\]
\[(\text{Wang, pass, Road2})\]

In this example, the lack of time information is prob-
lematic. None of these facts are wrong, but it is impossible for
Xiao Wang to pass both sections at the same time. Therefore
obviously temporal information will disambiguate some,
although not all facts usually require such metadata. For
example, the intersection of Road 1 and Road 2 will not
change over time.

In order to represent the spatiotemporal facts that
occurred at a specific time, the time \(t\) of the fact occurrence
is incorporated into the triples of the knowledge graph, and a
quadruple \((s, p, o, t)\) is introduced to describe the spatiotem-
poral facts. Definition 1 defines the triple \(tr\), and Definition 2
introduces a quadruple \(q\) containing time information.

**Definition 1** *(triple \(tr\)*) Let \(E\) as the entity set and \(R\) as the
relation set. A quadruple \(q\) containing time information is
represented as \(s, p, o, t \in E \times R \times E\), where \(s\) represents the
subject entity, \(o\) represents the object entity, and \(p\) represents
the relationship between the subject entity and the object
entity.

**Definition 2** *(quadruple \(q\)*) Let \(E\) as the entity set, \(R\)
as the relation set, and \(T\) as the time label set. A quad-
rule \(q\) containing time information is represented as
\((s, p, o, t) \in E \times R \times E \times T\), where \(s\) represents the subject
entity, \(o\) represents the object entity, \(p\) represents the relation-
ship between the subject entity and the object entity, and \(t\)
represents time information.

According to Definition 2, for the above example, we
can obtain the following quadruplets:

\((\text{Wang, pass, Road 1, 2020/9/10 10:17:00}),\]
\((\text{Wang, pass, Road 2, 2020/9/10 10:21:00})\]

3.2 Representation of Urban Traffic Knowledge
Graph

An urban traffic knowledge graph is a knowledge graph
that includes both temporal and spatial information. The
knowledge contained in the urban traffic knowledge graph
will change in time and space as compared with the usual
static knowledge graph. The urban traffic knowledge graph
can provide knowledge with temporal and spatial informa-
tion for subsequent tasks.

We combine the model of the road network layer with
the model of the trajectory layer to form a complete model
of the urban traffic knowledge graph. We define the con-
cepts of the urban traffic knowledge graph in Definition 3.

**Definition 3** *(urban traffic knowledge graph)* The urban traffic knowledge graph is denoted as
\(G = G_N \cup G_T = \{V, E, L, F\}\), where

- \(V = V_N \cup V_T\) represents the set of nodes in the urban
  traffic knowledge graph;
- \(E = E_N \cup E_T\) represents the set of relationships (edges)
between two nodes in the urban traffic knowledge graph;
- \(L = L_N \cup L_T = L_v \cup L_E\) represents the set of labels of all
  nodes and relationships in the urban traffic knowledge graph;
- \(F : F_N \cup F_T\) is the mapping function of all entities and
  relations in the urban traffic knowledge graph to the
  label set.

\(\odot\) Springer
In order to construct the urban traffic knowledge graph, the schema layer of the urban traffic knowledge graph needs to be constructed. The schema layer is the generalization and abstraction of knowledge, and the schema layer that builds the urban traffic knowledge graph is equivalent to establishing the ontology. The most basic ontologies include concepts, concept hierarchies, entities, entity attributes, relationships, relationship attributes, and attribute value types. The urban traffic knowledge graph we define includes the road network layer and the trajectory layer. The road network layer mainly includes entities related to the road network, attributes of entities, relationships, and attributes of relationships, while the trajectory layer mainly includes entities related to the trajectory and attributes of entities, relationships, and attributes of relationships. The constructed urban traffic knowledge graph pattern layer is shown in Fig. 2.

In our work, the data used are road network data and trajectory data. Therefore, the urban traffic knowledge graph we consider will be divided into two parts: the road network layer and the trajectory layer. The road network layer is a directed graph, which is composed of entities, relationships, attributes, and labels related to the road network in Nanjing. The road network layer contains spatial knowledge, which is similar to the traditional static knowledge graph. Therefore, a set of triples is used to form the road network layer. We define the concepts of the road network layer of the urban traffic knowledge graph in Definition 4.

**Definition 4** *(road network layer)* The urban traffic knowledge graph road network layer is denoted as $G_N = \{V_N, E_N, L_N, F_N\} = \{tr_1, tr_2, \ldots, tr_n\}$, where

- $V_N = V_r \cup V_s \cup V_j$ represents the set of road network entity nodes;
- $E_N$ represents the set of relationships (edges) between two road network entity nodes;
- $L_N = L_{NV} \cup L_{NE}$ represents the set of labels of road network entity nodes and relationships;
- $F_N : V_N \cup E_N \rightarrow L_N$ is the mapping function of the road network entity node and the relationship to the label set;
- $tr_1, tr_2, \ldots, tr_n$ is the set of triples that make up the road network layer of the urban traffic knowledge graph.

In Definition 4, $V_r$, $V_s$, and $V_j$ represent road entity nodes, road segment entity nodes, and road junction entity nodes, respectively. $L_N$ consists of $L_{NV}$ and $L_{NE}$, where $L_{NV}$ is the set label of all road entity vertices and $L_{NE}$ is the set label of all road entity relations. $F_N(V_N) : V_N \rightarrow L_{NV}$ is the mapping function from the road network entity node to the label of the road network entity node, and $F_N(E_N) : E_N \rightarrow L_{NE}$ is the mapping function from the road network entity relationship to the label of the road network entity relationship. We describe the road network layer in Definition 4. The triple (Road 1, intersect, Road 2) belongs to the road network layer. The triple is static and represents the spatial relationship between Road 1 and Road 2.

The trajectory layer consists of entities, relationships, attributes, and labels related to Nanjing taxi trajectories. The trajectory layer contains not only spatial knowledge, but also temporal knowledge, and the relationship between its two specific entities may change over time, which is the characteristic of the spatiotemporal knowledge graph, such that the set of quadruples is used to form the trajectory layer. Definition 5 defines the concepts of the trajectory layer of the urban traffic knowledge graph.
Definition 5 (trajectory layer) The urban traffic knowledge graph trajectory layer is denoted as
\[ G_T = \{V_T, E_T, L_T, F_T\} = \{q_1, q_2, ..., q_n\}, \]

where

- \(V_T\) represents the set of trajectory entity nodes;
- \(E_T = E_{TT} \cup E_{TN}\) represents the set of edges (relationships) between a trajectory entity node and another trajectory entity node or road network entity node;
- \(L_T = L_{TV} \cup L_{TE}\) represents the set of labels of trajectory entity nodes and relationships;
- \(F_T : V_T \cup E_T \rightarrow L_T\) is the mapping function of trajectory entity node and relationship to label set;
- \(q_1, q_2, ..., q_n\) is the set of quaternions that make up the trajectory layer of the urban traffic knowledge graph.

In Definition 5, \(E_{TT}\) and \(E_{TN}\), respectively, represent the encounter relationship between the trajectory entity nodes and the start, pass and end relationship between the trajectory and entity nodes mapped to the road network. \(L_T\) consists of \(L_{TV}\) and \(L_{TE}\), where \(L_{TV}\) is the set label of all trajectory entity vertices and \(L_{TE}\) is the set label of all trajectory entity relationships. \(F_T(V_T) : V_T \rightarrow L_{TV}\) is the mapping function of the label of the trajectory entity node to the trajectory entity node, and \(F_T(E_T) : E_T \rightarrow L_{TE}\) is the mapping function of the road network entity relationship to the label of the road network entity relationship. We describe the trajectory layer in Definition 5. The quadruple (Wang, pass, Road 1, “2020/9/10 10:17:00”) belongs to the trajectory layer. The quadruple carries time information and represents Wang’s movement trajectory.

Because of the large number of symbols in this section, the full name correspondence table of the symbol abbreviations describing the meaning of the symbols is given here, as shown in Table 4.

### 4 Construction of Urban Traffic Knowledge Graph

#### 4.1 Urban Traffic Data Acquisition

According to the model of urban traffic knowledge graph constructed in Sect. 3, this paper uses data with spatiotemporal characteristics, such as road network data and taxi data. We build the urban traffic knowledge graph based on the road network layer and the trajectory layer. Therefore, entity extraction and relation extraction need to be performed from the initial data to obtain the entities and relationships required in the road network layer and the trajectory layer in the urban traffic knowledge graph. The road network dataset includes road node data and road segment data. The road node data format is comma-separated values (CSV) and contains 23,682 road nodes. The road segment data format is CSV and contains 56967 road segments. Figure 3 shows the overall framework of the construction process of the urban traffic knowledge graph.

According to the content of Sect. 3, the road junction data are regarded as the road junction entity, and the road segment data are regarded as the road segment entity. The road network layer part of the urban traffic knowledge graph still lacks road entities. According to the road ID and road name information in the road segment entity, the road entities of the road network layer are extracted, and a total of 27880 road entities are extracted.

| Table 4 | Frequently used notations |
|---|---|
| Symbol | Meaning |
| s | Subject entity |
| p | Predicate |
| o | Object entity |
| t | Timestamp |
| tr | Triple |
| q | Quadruple |
| E | Entity set |
| R | Relation set |
| T | Time label set |
| G | Urban traffic knowledge graph |
| GN | Urban traffic knowledge graph road network layer |
| GT | Urban traffic knowledge graph trajectory layer |
| V | The set of nodes |
| E | The set of edges (relationships) in the urban traffic knowledge graph |
| L | The set of labels of all nodes and relationships |
| F | The mapping function of all entities and relations to the label set |
After having the road entity, it is necessary to extract the topological relationship and belonging relationship of the road network layer.

Firstly, link the road segment data file and the road junction data file according to the ID of the road junction to extract the relationship between the road junction and the road junction belonging to the same road segment, the relationship of the road junction belonging to the road segment and the relationship of the road segments include the road junction. The relationships between the road junction and the road junction belong to the same road segment contain 56,967 relationships, the relationships that road junction belong to road segments contain 113,843 relationships, and the relationships that road segments include road junction contain 113,843 relationships.

Then we link the road segment data file and the road data file and extract the relationship that the road segments belong to the road, the relationship that the roads include the road segment, and the relationship that the road segment and the road segment belong to the same road according to the ID of the road segment. Among them, there are 62,003 relationships in which road segments belong to roads, 62,003 relationships in which roads include road segments, and 365,876 relationships in which road segments belong to the same road.

Finally, the road segment data file is self-linked, and the intersect relationship between the road segment is extracted by judging whether the two road segments contain a common road junction. The intersect relationship between the road segment contains 355804 relationships, and the attribute in the relationship contains the intersection point where the two road segments intersect.

The data format of the taxi dataset is CSV, which contains the trajectory data of one thousand taxis, with a total of 3,242,557 trajectory points.

In the trajectory layer part of the urban traffic knowledge graph, the trajectory data of the taxi is regarded as the trajectory entity, and the brute force algorithm is used to calculate the encounter relationship of the trajectories of the taxi. The brute force algorithm approaches the identification of a potential collision by iteratively traversing the entirety of the lists representing the coordinates corresponding to the two UIIDs under examination. For each coordinate of the first list, the Euclidean distance to every coordinate of the second list is calculated and if this is smaller than the specified ‘spatialEpsilon’ then their timestamps are checked against the provided ‘temporalEpsilon’ and their floors are checked for equality. After calculation, a total of 483995 encounter relationships between trajectories are obtained. In addition, with the help of the mapping function, the trajectory data of the taxi is mapped to the road network, and the start, pass and end relationship between the trajectory and the road segment is extracted, which can, respectively, show which road segment the taxi departs from. Information on which road segments the taxi passes through and which road segment the taxi finally reaches. A total of 1000 start relations, 3240557 pass relations, and 1000 end relations were obtained. We connect the road network layer and the trajectory layer by the extracted start, pass and end relations.

### 4.2 Knowledge Extraction from Urban Traffic Data

The knowledge extraction from urban traffic data can be divided into three steps: entity classification, relation extraction, and attribute extraction.

#### 4.2.1 Entity Classification

According to the definition of the pattern layer of the urban traffic knowledge graph in Sect. 3, the urban traffic knowledge graph consists of the road network layer and the trajectory layer. The road network layer includes three entities: road entity, road segment entity, and road junction entity. The entity in the trajectory layer is the trajectory entity. As shown in Table 5.
Table 5  Entity classification

| Layer               | Entity name          |
|---------------------|----------------------|
| Road network layer  | Road                 |
|                     | Road segment         |
|                     | Road junction        |
| Trajectory layer    | Trajectory           |

Table 6  Relationship of the urban traffic knowledge graph

| Relationship       | Entity A            | Entity B            |
|--------------------|---------------------|---------------------|
| SameRoadSegment    | Road junction       | Road junction       |
| BelongsTo          | Road junction       | Road segment        |
| Include            | Road segment        | Road junction       |
| Intersect          | Road segment        | Road segment        |
| SameRoad           | Road segment        | Road segment        |
| BelongsTo          | Road segment        | Road                |
| Include            | Road                | Road segment        |
| Encounter          | Trajectory          | Trajectory          |
| Start              | Trajectory          | Road segment        |
| Pass               | Trajectory          | Road segment        |
| End                | Trajectory          | Road segment        |

Table 7  Attribute of trajectory entity

| Entity            | No | Attribute name  |
|-------------------|----|-----------------|
| Trajectory        | 1  | GPSTime         |
|                   | 2  | GPSLongitude    |
|                   | 3  | GPSLatitude     |
|                   | 4  | GPSSpeed        |
|                   | 5  | GPSDirection    |
|                   | 6  | PassengerState  |
|                   | 7  | ReadFlag        |
|                   | 8  |CreateDate      |

Table 8  Attribute of road entity

| Entity   | No | Attribute name  |
|----------|----|-----------------|
| Road     | 1  | Road name       |
|          | 2  | Road type       |

Table 9  Attribute of segment entity

| Entity       | No | Attribute name   |
|--------------|----|------------------|
| Road segment | 1  | Start point      |
|              | 2  | End point        |
|              | 3  | Highway          |
|              | 4  | One way          |
|              | 5  | Length           |
|              | 6  | Speed/kph        |
|              | 7  | Travel time      |
|              | 8  | Edge centrality  |
|              | 9  | Geometry         |
|              | 10 | Lanes            |
|              | 11 | Name             |
|              | 12 | Ref              |
|              | 13 | Bridge           |
|              | 14 | Junction         |
|              | 15 | Maxspeed         |
|              | 16 | Tunnel           |
|              | 17 | Width            |
|              | 18 | Access           |

Table 10  Attribute of junction entity

| Entity      | No | Attribute name   |
|-------------|----|------------------|
| Road junction | 1  | Longitude        |
|             | 2  | Latitude         |
|             | 3  | Street count     |
|             | 4  | Node centrality  |
|             | 5  | Highway          |
|             | 6  | Geometry         |

4.2.2 Relationship Extraction

According to the definition of the schema layer of the urban traffic knowledge graph in Sect. 3, a total of nine relationships are defined. As shown in Table 6.

4.2.3 Attribute Extraction

In the schema layer of the urban traffic knowledge graph, attributes include attributes of entities and attributes of relationships. Take the trajectory entity as an example. Table 7 shows the attributes of the trajectory entity.

Table 8 shows the attributes of the road entity.
Table 9 shows the attributes of the segment entity.
Table 10 shows the attributes of the junction entity.

4.3 Organization and Management of Urban Traffic Knowledge

The knowledge graph can be stored based on table structure and graph structure. Currently recognized graph data models in the industry include attribute graph, resource description framework (RDF), and triple hypergraph, the first two of which are widely used in graphs Database products. Therefore, knowledge graphs can use relational databases and graph databases (such as Neo4j, Jena, and Virtuoso). In contrast, the former has lower search efficiency, while the latter is more conducive to data reading, writing, storage, and query. After our research, the graph database Neo4j has
good storage performance, and due to a large number of users, it has relatively complete learning routes and technical documentation support, which can meet our needs for urban traffic knowledge graph storage. Therefore, the graph database Neo4j is used as the local persistent storage to improve the efficiency of reading, writing, storage, and query.

5 TKG_Sub: Star Subgraph Query on Urban Traffic Knowledge Graph

Based on the model of the urban traffic knowledge graph proposed in Sect. 3 and the urban traffic knowledge graph constructed in Sect. 4, this section proposes a query method on the urban traffic knowledge graph. Figure 4 shows the overall framework of the urban traffic knowledge star subgraph query algorithm TKG_Sub.

5.1 Urban Traffic Star Subgraph Query

In order to perform star subgraph query on the urban traffic knowledge graph, we combine the concept of the urban traffic knowledge graph to define the urban traffic query graph. Definition 6 introduces the concept of urban traffic query graph.

Definition 6 (urban traffic query graph) The urban traffic query graph is denoted as \( G_q = (V_q, E_q, L_q, F_q) \), where

- \( V_q \) represents the set of nodes in the spatiotemporal query graph;
- \( E_q \) represents the set of edges (relationships) between two nodes in the spatiotemporal query graph;
- \( L_q \) represents the set of labels of all nodes and relationships in the spatiotemporal query graph;
- \( F_q \) is the mapping function of all entities and relations in the spatiotemporal query graph to the label set.

In Definition 6, the meanings of \( V_N, V_T, E_N, E_T, L_N, L_T, L_V \) and \( L_E \) are the same as in Definition 4. \( F_q(V_q) : V_q \rightarrow L_V \) is the mapping function from nodes to node labels, and \( F_q(E_q) : E_q \rightarrow L_E \) is the mapping function from relations to relation labels.

The star subgraph query of the urban traffic knowledge graph can be modeled as the classical subgraph isomorphism problem in graph theory. For a given spatiotemporal query graph \( G_q \), if there is at least one subgraph \( g \) in the urban traffic knowledge graph \( G \), such that \( G_q \) is isomorphic to \( g \), then \( G_q \) is considered subgraph isomorphic to \( G \). The concept of subgraph query of urban traffic knowledge graph is shown in Definition 7.

Definition 7 (subgraph query of urban traffic knowledge graph) For the query graph \( G_q = (V_q, E_q, L_q, F_q) \) and a certain subgraph \( g = (V', E', L', F') \), \( g \) is said to be a match of \( G_q \) if and only if there exists a function \( f \) that satisfies the following conditions:

- For the vertex set \( V_q \) of the query graph \( G_q \) satisfies \( V_q \subseteq V \), i.e. the vertex set \( V_q \) of the query graph \( G_q \) is a proper subset of the vertex set \( V \) of the urban traffic knowledge graph \( G \);
- For the edge set \( E_q \) of the query graph \( G_q \) satisfies \( E_q \subseteq E \), i.e. the edge set \( E_q \) of the query graph \( G_q \) is the proper subset of the edge set \( E \) of the urban traffic knowledge graph \( G \);
- For any vertex \( v \in V_q \) satisfy \( f(v) \in V' \), and there is \( F_q(v) = F'(f(v)) \);
- For any two nodes \( v_1 \) and \( v_2 \) in \( V_q \), there is an edge \( e(v_1, v_2) \in E_q \) between \( v_1 \) and \( v_2 \) if and only if there is \( e(f(v_1), f(v_2)) \in E' \) and \( F_q(e(v_1, v_2)) = F'(e(f(v_1), f(v_2))) \).

In Definition 7, the injective function \( f \) is used to find subgraphs in the urban traffic knowledge graph with the same structure and features as the urban traffic query graph.

5.2 Query Pattern

Since we construct the urban traffic knowledge graph in the previous section, we need to consider the corresponding application background of the urban traffic knowledge graph in the subgraph query. According to possible application scenarios, we mainly consider two query modes, namely star subgraph query pattern with a center node and star subgraph query pattern with multiple star center nodes.

For the star subgraph query pattern, it is also one of the basic query patterns in the subgraph query. The star-shaped graph consists of a star-shaped central node and its adjacent nodes, and the star-shaped central node can fully describe the characteristic information of adjacent nodes. In the urban traffic knowledge graph, there are a large number of applications that correspond to the star subgraph query mode. For example, if we want to find a moving object and all other moving objects that have contact with that moving object, then we will get a star query subgraph from the urban traffic knowledge graph. The knowledge meaning represented by
An Efficient Algorithm of Star Subgraph Queries on Urban Traffic Knowledge Graph

this subgraph is that of a person and everyone else he has come into contact with. Such knowledge is very useful in scenarios such as assisting epidemic prevention and control. Figure 5 is an example of a star subgraph query pattern with a center node and four adjacent nodes.

In addition to star subgraphs with a single star center node, there also exists star subgraphs with multiple star center nodes. The star subgraph consists of multiple star center nodes and their adjacent nodes. The star subgraph with multiple star center nodes can describe the dense area of intersections in the road network layer in the urban traffic knowledge graph, and it can express the knowledge of density and spatial structure in urban roads. In addition, combined with the trajectory knowledge of the trajectory layer, it can also express the knowledge about traffic flow, which is very helpful for user travel, route planning, urban planning, and other applications. Figure 6 is an example of a star subgraph query pattern with three star center nodes and six adjacent nodes.

**5.3 Determining the Node Query Candidate Area**

Considering the urban traffic query graph, the matching order of the query nodes needs to be calculated during the subgraph query process. In order to clarify the matching order of the query nodes, we find the node query candidate area. Definition 8 introduces the concept of candidate area.

**Definition 8** (candidate area) For the matching order, \( D(u) \) is the query candidate area of node \( u \) in the urban traffic knowledge graph, where \( D(u) \) contains all data nodes that may match \( u \). Node \( u \) and any node \( v \) in \( D(u) \) should meet the following conditions:

1. \( \text{deg}(u) \leq \text{deg}(v) \);
2. \( \text{deg-in}(u) \leq \text{deg-in}(v) \);
3. \( \text{deg-out}(u) \leq \text{deg-out}(v) \).

The \( \text{deg} \) function \( \text{deg}(u) \) represents the degree of node \( u \), where \( \text{deg-in}(u) \) represents the in-degree of node \( u \), and \( \text{deg-out}(u) \) represents the out-degree of node \( u \). Out-degree and in-degree are numbers of outcoming and incoming edges from a node. If \( \text{deg-in}(u) \leq \text{deg-in}(v) \) and \( \text{deg-out}(u) \leq \text{deg-out}(v) \) are satisfied for nodes \( u \) and \( v \), then \( \text{deg}(u) \leq \text{deg}(v) \). In addition, if there is an edge connecting the query nodes \( u_1 \) and \( u_2 \) in two urban traffic query graphs, a node \( v_1 \) in the query candidate area \( D(u_1) \) of \( u_1 \) must be adjacent to a node \( v_2 \) in the query candidate area \( D(u_2) \) of \( u_2 \), i.e., there is an edge between \( v_1 \) and \( v_2 \). The rule is called the principle of arc consistency. This means that if a node in the urban traffic knowledge graph exists in the query candidate area of node \( u \) in the urban traffic query graph but does not meet the principle of arc consistency, then it should be removed from \( D(u) \).

**5.4 Calculation of the Matching Order**

For the star subgraph matching of the urban traffic query graph, the matching order of the query nodes needs to be calculated during the star subgraph query process, and in order to clarify the matching order of the query nodes, the first query node should be determined. The star subgraph of the urban traffic knowledge graph is queried by the star subgraph query algorithm, and the first query node is selected according to the following rules:

1. The node in the smallest query candidate area (i.e., the least number of nodes in the query candidate area) is selected as the first query node. When the query candidate area with two or more nodes is the smallest, adopt method 2 to select these nodes;
2. The node with the largest degree is selected as the first node. When there are two or more query nodes with the same maximum degree, the first node can be selected using the method in 3;
3. The node with the largest outdegree is selected as the first node. When there are two or more nodes with the...
same maximum outdegree, any node will be selected as the first query node.

For $n$ query nodes in the urban traffic query graph, after the first query node is determined, the first query node is added to the sorted node, and then the remaining $n-1$ query nodes are determined according to the sorted nodes. Among the remaining query nodes to be sorted, the node with the highest correlation with the sorted node should be preferentially added to the sorted node. An approach is adopted to sort subsequent query nodes. $\zeta_i = \{u_1, u_2, ..., u_i\}$ denotes a sorted set of nodes consisting of $i$ nodes, where $i \leq n$. $\eta$ is the set of remaining query nodes. In order to select the ordered next node from the remaining query nodes, three sets of candidate query nodes are defined:

1. $V_{u,adj}$: The set of query nodes that belong to $\zeta_i$ and are adjacent to $u$;
2. $V_{u,boun}$: The set of query nodes belonging to $\zeta_j$ that are adjacent to $u$ and adjacent to at least one other node in $\eta$;
3. $V_{u,sep}$: The set of query nodes that are adjacent to $u$ but not belonging to $\zeta_i$ and not adjacent to nodes in $\zeta_j$.

Select the next query node from the remaining query nodes with the following rules: Firstly, choose the query node with the largest value of $|V_{u,adj}|$. Secondly, Select the query node with the largest value of $|V_{u,boun}|$ if the value of $|V_{u,adj}|$ is the same; Thirdly, Select the query node with the largest value of $|V_{u,sep}|$ if the value of $|V_{u,adj}|$ is the same; Finally, Select any of the nodes if the value of $|V_{u,sep}|$ is the same.

Referring to Fig. 6 as an example, according to the rules for selecting the first query node, since $v_2$ is located in the smallest query candidate area and has the largest degree, we choose $v_3$ as the first node, then $\zeta_1 = \{v_2\}$, $\eta = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$. When selecting the next query node, consider

$$V_{v_1,adj} = V_{v_1,adj} = V_{v_2,adj} = V_{v_3,adj} = \{v_2\}$$

and

$$V_{v_2,adj} = V_{v_3,adj} = V_{v_4,adj} = \emptyset$$

such that $v_1,v_3,v_2,v_3,v_6$, and $v_3$ can consider the next query node among them. Considering

$$V_{v_1,boun} = V_{v_1,boun} = V_{v_2,boun} = V_{v_3,boun} = \{v_2\}$$

we still cannot determine the next query node. Then use rule three to proceed with determination. Because

$$V_{v_4,sep} = \{v_8, v_9\}, V_{v_5,sep} = \{v_4\} V_{v_6,sep} = V_{v_7,sep} = V_{v_7,sep} = \emptyset$$

$|V_{v_{i,sep}}|$ is the largest. From this, it can be determined that $v_3$ is the next query node. After updating $\zeta_1$ and $\eta$, there are $\zeta_2 = \{v_2, v_3\}$ and $\eta = \{v_1, v_4, v_5, v_6, v_7, v_8, v_9\}$. Then select the next query node in turn and the final query sequence is $\{v_2, v_3, v_1, v_4, v_5, v_6, v_7, v_8, v_9\}$.

In order to find matching results with the highest efficiency and reduce the useless traversal of the graph, the first query node should select the query node with the largest out-degree. The order of the remaining $n-1$ query node is determined by the degree of association with the sorted node. The importance of this classification is to consider the proximity of the relationship between nodes. To facilitate the creation of best query results.

### 5.5 The Query Process of Star Subgraph Query

Star subgraph query of urban traffic knowledge graph is the process of finding star subgraphs isomorphic to urban traffic query graph in the urban traffic knowledge graph. To speed up this process, we target the temporal features of the urban traffic knowledge graph and use a filtering mechanism to exclude queries with a time range outside the data graph. In the urban traffic knowledge graph, there are a number of entities and relationships with temporal characteristics, and the temporal attributes of these entities and relationships together constitute the time interval of the urban traffic knowledge graph. Similarly, the time attributes of entities and relationships in the urban traffic query graph together constitute the time interval of the urban traffic query graph. Only when the time interval of the urban traffic query graph $G_q$ is a subset of the time interval of the urban traffic knowledge graph $G$, there may be star subgraphs in $G$ that match $G_q$. For example, when the time interval of the urban traffic knowledge graph $G$ is [2020/09/12 0:00:00, 2020/10/12 0:00:00]. If the time interval of the query graph $G_q$ is [2020/09/12 6:01:00, 2020/09/15 09:54:00], since the time interval of $G_q$ is a subset of the time interval of $G$, therefore, in $G$ There may be star subgraphs matching $G_q$. If the time interval of the query graph $G_q$ is [2020/09/10 13:21:00, 2020/10/13 14:36:00], the time interval of $G_q$ is not a subset of the time interval of $G$. There is no star subgraph in $G$ that matches $G_q$. This query is useless and can directly return empty results.

In Definition 7, the injective function $f$ is used to find star subgraphs in the urban traffic knowledge graph with the same structure and features as the urban traffic query graph. In order to find the isomorphic star subgraph corresponding to the urban traffic query graph $G_q$ in the urban traffic knowledge graph $G$ to complete the query based on the urban traffic knowledge graph, a star subgraph query algorithm TKG_Sub based on the urban traffic knowledge graph is proposed. Algorithm 1 outlines the overall process.
An Efficient Algorithm of Star Subgraph Queries on Urban Traffic Knowledge Graph

Algorithm 1 TKG_Sub(Urban Traffic Knowledge Star Subgraph Query Algorithm)

**Input:** urban traffic knowledge graph $G$, urban traffic query graph $G_q$

**Output:** the set $R$ of star subgraphs in $G$ that match $G_q$

1. $T_G \leftarrow \text{GetTimeSpan}(G)$
2. $T_Q \leftarrow \text{GetTimeSpan}(G_q)$
3. if $T_Q \subseteq T_G$ then
   4. $u \leftarrow \text{ChooseFirstVertex}(G_q)$
   5. $D(u) \leftarrow \text{GetCandidate}(G, u)$
   6. if $D(u) \neq \emptyset$ then
      7. $\text{Ord} \leftarrow \text{OrderMatch}(G_q)$
      8. for each $v \in D(u)$ do
         9. \text{SubgraphMatch}$(G, G_q, \text{Ord}, u, v)$
      10. $\text{UpdateState}(u, v, R)$
     end for
   end if
end if

The key point of the star subgraph matching process of TKG_Sub is the recursive process, which takes backtracking as the basic idea. Algorithm 2 presents the star subgraph search algorithm $\text{SubgraphMatch}$.

Algorithm 2 $\text{SubgraphMatch}$(Subgraph Search Algorithm)

**Input:** urban traffic knowledge graph $G$, urban traffic query graph $G_q$, matching order $\text{Ord}$, data node $v$, query node $u$

**Output:** the star subgraph $g$ in $G$ that match $G_q$

1. if $V_g = V_q$ and $E_g = E_q$ then
   2. return $g$
3. else
   4. $u' \leftarrow \text{NextQueryVertex}()$
   5. $C(u') \leftarrow \text{Neighbor}(f(u)) \cap \text{GetCandidate}(G, u')$
   6. if $C(u') \neq \emptyset$ then
      7. for each $v' \in C(u')$ do
         8. if $c(u', u) \in E_q$ and $c(v', v) \notin E$ then
            9. $C(u') \leftarrow C(u') \setminus \{v'\}$
         end if
      end for
   end if
   12. for each $v' \in C(u')$ such that $v'$ is not matched do
      13. $\text{Check}(u', v')$
      14. $\text{SubgraphMatch}$(G, $G_q$, $\text{Ord}$, $u'$, $v'$)
      15. $\text{UpdateState}(u', v', g)$
   end for
17. end if
18. end if

The complexity analysis In Algorithm 1, the main source of time complexity is the function $\text{SubgraphMatch}$, which is the main component of Algorithm 2. We mainly analyze the time complexity of the function $\text{SubgraphMatch}$. 

In the process of query, obtain the time interval of $G$ and $G_q$ through the $\text{GetTimeSpan}$ function. If the time interval of $G_q$ is not a subset of the time interval of $G$, it means that there is no star subgraph in $G$ that matches $G_q$ (lines 1-2). Then, use the $\text{ChooseFirstVertex}$ function to choose the first query node in $G_q$ (line 4). Next, get the candidate area of the node through the $\text{GetCandidate}$ function (line 5). If the candidate area is not empty, sort the remaining n-1 query nodes in $G_q$. Use the $\text{SubgraphMatch}$ function in turn for each query node in the candidate area. When a query node $u$ is found in the query graph and a matching data node $v$ is found in the data graph, the set $R$ is updated through the update function $\text{UpdateState}$, and finally a complete set $R$ of all star subgraphs in $G$ that match $G_q$ is returned, where $R$ contains matching star subgraphs of $G_q$ in $G$. 

The complexity analysis In Algorithm 1, the main source of time complexity is the function $\text{SubgraphMatch}$, which is the main component of Algorithm 2. We mainly analyze the time complexity of the function $\text{SubgraphMatch}$. 

 Springer
Let \( n \) be the vertex size of the query graph \( q \). We can know that the overall steps of the algorithm can be divided into two main steps. First, we find and add all vertices of the query graph \( q \) to the subgraph \( g \), which requires \( O(n) \) time complexity. Then each time a vertex is added to the subgraph \( g \), we sort the adjacent node sets of the added vertex and select the next vertex to be added. This step is performed by the function \( \text{NextQueryVertex} \) (line 4) and requires \( O(n \cdot \log n) \) time complexity. The time complexity of the algorithm is \( O(n^2 \cdot \log n) \).

### 6 Experimental Evaluation

#### 6.1 Environment

The query method based on the urban traffic knowledge graph is implemented in Java. All experiments were performed under the 64-bit operating system of Windows 10, using an x64-based processor. The processor is Intel(R) Core(TM) i5-10500 CPU @ 3.10GHz, and RAM is 8.00 GB.

#### 6.2 Datasets

In this paper, data with spatial and temporal characteristics such as Nanjing Road Network Data and Nanjing Taxi Data Set are collected. The Nanjing Road Network Data Set comes from OpenStreetMap, an online map collaboration project. The Nanjing road network dataset includes road junction data, road segment data, and POI data. The road node data format is CSV and contains 23,682 road junctions. The road segment data format is CSV and contains 56,967 road segments. The POI data format is CSV and contains 147,046 POI. The data format of the Nanjing taxi dataset is CSV, which contains the trajectory data of one thousand taxis, with a total of 3,242,557 trajectory points. This paper builds the urban traffic knowledge graph based on the road network layer and the trajectory layer. Therefore, entity extraction and relation extraction need to be performed from the initial data to obtain the entities and relationships required in the road network layer and the trajectory layer in the urban traffic knowledge graph. After obtaining the entities and relationships required for the road network layer and the trajectory layer in the urban traffic knowledge graph, the urban traffic knowledge graph is constructed. Experiments on star subgraph query are carried out on the constructed urban traffic knowledge graph, and the performance of star subgraph query algorithm is measured by comparative experiments. Table 11 shows the statistics of the datasets.

| Data sets  | Size   |
|------------|--------|
| Road junction | 23,682 |
| Road segment  | 56,967 |
| Taxi         | 3,242,557 |
| POI          | 147,046 |

#### 6.3 Experimental Setup

According to the description of the query pattern in the previous section, we divide the experiments into two aspects and conduct experiments on the efficiency of the star subgraph query method.

The first aspect is for the star query pattern with one central node, which is the basis for the star query pattern. Three different sets of star queries are set up. In each set of queries, the central node and adjacent nodes are the moving entities of the trajectory layer in the urban traffic knowledge graph, and the edges between them are the encounter relationships between the moving entities. The specific contents of the three sets of star queries are as follows:

1. The star query graph contains one central node, and three adjacent nodes, for a total of four nodes;
2. The star query graph contains one central node, and four adjacent nodes, for a total of five nodes;
3. The star query graph contains one central node, and five adjacent nodes, for a total of six nodes.

The second aspect is for the star query mode with a plurality of central nodes, which is a more complex star query mode. In the urban traffic knowledge graph, the star subgraphs with multiple star center nodes often correspond to the dense area of the intersection. The central node is the intersection entity of the road network layer in the urban traffic knowledge graph, and the adjacent nodes are the urban traffic knowledge graph. The road segment entities of the middle road network layer and the edges between them are the relationship that the intersection belongs to the road segment. As the number of central nodes increases, the number of corresponding star subgraphs in the urban traffic knowledge graph will decrease. The proportion of star subgraphs with more than five central nodes in the urban traffic knowledge graph is very small and can be ignored. In a star subgraph with multiple central nodes, the maximum number of central nodes is seven. Figure 7 shows the distribution of the proportion of star subgraphs with central nodes 2, 3, 4, and 5 in the urban traffic knowledge graph. Four different sets of star queries are set up according to this distribution in Fig. 7. In each set of queries, the central node is the road junction entity of the road network layer in the urban traffic knowledge graph, the adjacent nodes are the road segment...
entities of the road network layer in the urban traffic knowledge graph, and the edges between them are the relationship that road junction belong to the road segment. The specific contents of the four sets of star queries are as follows:

1. The star query graph contains two central nodes, eight adjacent nodes, and a total of ten nodes;
2. The star query graph contains three central nodes, eleven adjacent nodes, and a total of fourteen nodes;
3. The star query graph contains four central nodes, eighteen adjacent nodes, and a total of twenty-two nodes;
4. The star query graph contains five central nodes, twenty-two adjacent nodes, and a total of twenty-seven nodes.

We identify two categories of query graphs according to query patterns. The first category is a star subgraph with one central node, and the second category is a star subgraph with multiple central nodes. For the first category, we query a moving object and all other moving objects that have contact with that moving object. We find a moving object in the datasets that matches our experimental setup. Then we use that object and its connected nodes as the query graph. For the second category, we look for areas with dense intersections in the road network layer, and take a star subgraph with three central nodes which found in the datasets as an example. Firstly, we find the first center point A arbitrarily, and select the center point B adjacent to the center point A. It is necessary to satisfy the condition that the center point C adjacent to the center point B is adjacent to the center point A, then we determine the star subgraph with the three central nodes.

In addition, we compare TKG_Su b with the basic Ullmann algorithm [24] in the subgraph query algorithm, and the relatively advanced MPMatch algorithm [25] and Fast algorithm [26] within three years through comparative experiments. The Ullmann algorithm is mainly composed of two steps: a simple enumeration algorithm of subgraph isomorphism and an algorithm using a refinement procedure. The MPMatch algorithm is mainly used to perform subgraph matching in parallel. The algorithm divides the matching process into two stages: subtask generation and subtask processing. The FAST algorithm is based on a field programmable logic gate array, which uses the CPU-FPGA co-design framework. Test query efficiency by comparing the query response time of different algorithms for querying the same query graph.

6.4 Experimental Results

In this subsection, the experimental results are divided into algorithm effect and algorithm performance analysis.

6.4.1 The Effect of the Algorithms

In order to reduce the chance of experimentation, each set of query samples was run ten times, and the results were averaged. The experimental results are shown in Figs. 8 and 9. Figure 8 describes the query response time for a star query with one central node, and Fig. 9 describes the query response time for a star query with multiple central nodes.

As shown in Fig. 8. The query response time of a star query pattern with a central node, three adjacent nodes, and a total of four nodes is 0.84 s. The query response time of a star query pattern with a central node, four adjacent nodes, and a total of five nodes is 0.87 s. The query response time of a star query pattern with a central node, five adjacent nodes, and a total of six nodes is 0.98 s. It can be seen that as the number of nodes and edges in the query graph increases,
the time spent by the query increases at a slow rate. The main reason is that in the query process, it takes more time to match query subgraphs with more nodes in the urban traffic knowledge graph.

In Fig. 9. The query time of a star query pattern with two central nodes, eight adjacent nodes, and a total of ten nodes is 1.94 s. There are three central nodes, eleven adjacent nodes, and the query time of a star query pattern of fourteen nodes in total is 2.14 s. The query time of a star query pattern with four central nodes, eighteen adjacent nodes, and a total of twenty-two nodes is 2.84 s. With five central nodes, twenty-two adjacent nodes, and a total of twenty-seven nodes, the query time of a star query pattern is 3.92 s. As the number of nodes and edges in the query graph increases, the time spent in the query increases at a moderate rate. The main reason is that the more central nodes in the query graph, the more vertices and edges need to be compared during the query process, and the query algorithm needs more time. In addition, since the spatiotemporal data in the query graph have more attribute information and higher complexity than the non-spatiotemporal data, more complex matching is required in the query process.

6.4.2 The Analysis of the Algorithms

In the analysis part, TKG_Sub is compared with the basic Ullmann algorithm in the subgraph query algorithm and the relatively advanced MPMatch algorithm and Fast algorithm within three years, using the query response time (the time required for the operation to complete) as the matching indicator of efficiency.

As shown in Figs. 10 and 11, when the query graph is a star query graph with one central node, experiments are performed on star query graphs with total nodes 4, 5, and 6. The experimental results show that the query response time of the Ullmann algorithm increases faster when the number of query nodes increases, and the growth rate exceeds that of the other three algorithms. The query time of TKG_Sub, MPMatch and Fast all increase at a slow rate, but the query efficiency of TKG_Sub is higher, indicating that the algorithm performs well on the basic star query on the urban traffic knowledge graph.

As shown in Figs. 12 and 13, when the query graph is a star query graph with a plurality of central nodes, experiments are performed on the star query graph with central nodes 2, 3, 4, and 5. Experimental results show that the query response time of all algorithms increases. However, the query time of TKG_Sub grows at a smaller rate than other methods, which means that our algorithm is
more reliable. It can be observed that in the case of a star query with five central nodes, twenty-two adjacent nodes, and a total of twenty-seven nodes, TKG_Sub outperforms MPMatch and Fast in terms of efficiency by 20% and 30%, respectively, far better than Ullmann, reaching 93%.

We write a program to simulate query graphs. The program randomly selects the central vertex and its adjacent vertices in the urban traffic knowledge graph. The specific contents of the three groups of simulated star queries are as follows:

1. The star query graph contains 20 central nodes, 34 adjacent nodes, and a total of 54 nodes;
2. The star query graph contains 50 central nodes, 98 adjacent nodes, and a total of 148 nodes;
3. The star query graph contains 100 central nodes, 188 adjacent nodes, and a total of 288 nodes.

As shown in Figs. 14 and 15, we conduct experiments on three simulated star query graphs. The experimental results show that under the condition of the simulated query graph with 288 nodes, TKG_Sub has the minimum query response time and Ullmann has the maximum query time. In a star query graph with 100 central nodes, 188 adjacent nodes, and a total of 288 nodes, the efficiency of TKG_Sub is 21.5% and 32.1% higher than MPMatch and Fast, respectively, and 95% higher than Ullmann.

Fig. 12 Experimental comparison of TKG_Sub, MPMatch and Fast on the star query mode with multiple central nodes

Fig. 13 Experimental comparison of TKG_Sub and Ullmann on the star query mode with multiple central nodes

Fig. 14 Experimental comparison of TKG_Sub, MPMatch and Fast in simulating query graph

Fig. 15 Experimental comparison of TKG_Sub and Ullmann on the star query mode with multiple central nodes
7 Conclusion

In this paper, we use road network data and trajectory data to extract urban traffic knowledge from them, and construct an urban traffic knowledge graph. Then we propose a star subgraph query algorithm on the urban traffic knowledge graph. By using a filtering mechanism to exclude queries whose time range is beyond the data graph, the query efficiency on the urban traffic knowledge graph with time features is improved. Experiments show that the proposed method has better performance on the urban traffic knowledge graph. In future work, we plan to add a time index based on the preliminary time filtering mechanism to improve query efficiency.

Acknowledgements This work is supported by National Natural Science Foundation of China (61972198), Natural Science Foundation of Jiangsu Province of China (BK20191273).

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Tao Sun, Jianqiu Xu and Caiping Hu. The first draft of the manuscript was written by Tao Sun and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding National Natural Science Foundation of China (61972198), Natural Science Foundation of Jiangsu Province of China (BK20191273).

Availability of data and material Yes, available.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Ruan S, Long C, Bao J, Li C, Yu Z, Li R, Liang Y, He T, Zheng Y (2020) Learning to generate maps from trajectories. In: Proceedings of the AAAI conference on artificial intelligence vol 34, pp 890–897
2. Singhal A (2012) Introducing the knowledge graph: things, not strings. Official Google Blog 5:16
3. Bollacker K, Evans C, Paritosh P, Sturte T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on management of data, pp 1247–1250
4. Lehmann J, Isele R, Jakob M, Jentzsch A, Kontokostas D, Mendes PN, Hellmann S, Morsey M, Van Kleef P, Auer S et al (2015) Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web 6(2):167–195
5. Ciffolilli A (2003) Phantom authority, self-selective recruitment and retention of members in virtual communities
6. Suchanek FM, Kasneci G, Weikum G (2007) Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web, pp 697–706
7. Niu X, Sun X, Wang H, Song S, Qi G, Yu Y (2011) Zhishi. me: weaving chinese linking open data. In: International semantic web conference. Springer, pp 205–220
8. Xu B, Xu Y, Liang J, Xie C, Liang B, Cui W, Xiao Y (2017) Cn-dbpeda: a never-ending Chinese knowledge extraction system. In: International conference on industrial, engineering and other applications of applied intelligent systems. Springer, pp. 428–438
9. Wang Z, Li J, Wang Z, Li S, Li M, Zhang D, Shi Y, Liu Y, Zhang P, Tang J (2013) Xlore: a large-scale english-chinese bilingual knowledge graph. In: International semantic web conference (Posters and Demos), vol 1035, pp 121–124
10. Li Y, Qian B, Zhang X, Liu H (2020) Knowledge guided diagnosis prediction via graph spatial-temporal network. In: Proceedings of the 2020 SIAM international conference on data mining. SIAM, pp 19–27
11. Del Mondo G, Peng P, Gensel J, Claramunt C, Lu F (2021) Leveraging spatio-temporal graphs and knowledge graphs: Perspectives in the field of maritime transportation. ISPRS Int J Geo Inf 10(8):541
12. Huang Y, Yin P, Zhou G, Liu P, Tang Y, Li W (2020) Construction of public safety knowledge graphs. In: 2020 International conference on computer, information and telecommunication systems (CITS)
13. Zhu C, Chen M, Fan C, Cheng G, Zhan Y (2020) Learning from history: modeling temporal knowledge graphs with sequential copy-generation networks. arXiv preprint arXiv:2012.08492
14. Trivedi R, Dai H, Wang Y, Song L (2017) Know-evolve: deep temporal reasoning for dynamic knowledge graphs. In: International conference on machine learning. PMLR, pp 3462–3471
15. Jin W, Qu M, Jin X, Ren X (2019) Recurrent event network: autoregressive structure inference over temporal knowledge graphs. arXiv preprint arXiv:1904.05530
16. Han Z, Ding Z, Ma Y, Gu Y, Tresp V (2021) Temporal knowledge graph forecasting with neural ode. arXiv preprint arXiv:2101.05151
17. Xiao C, Sun L, Ji W (2020) Temporal knowledge graph incremental construction model for recommendation. In: Asia-pacific web (apweb) and web-age information management (waim) joint international conference on web and big data. Springer, pp 352–359
18. Zhuang C, Yuan NJ, Song R, Xie X, Ma Q (2017) Understanding people lifestyles: construction of urban movement knowledge graph from gps trajectory. In: IJCAI, pp 3616–3623
19. Chen J, Ge X, Li W, Peng L (2021) Construction of spatiotemporal knowledge graph for emergency decision making. In: IEEE international geoscience and remote sensing symposium IGARSS. IEEE, pp 3920–3923
20. Tan J, Qiu Q, Guo W, Li T (2021) Research on the construction of a knowledge graph and knowledge reasoning model in the field of urban traffic. Sustainability 13(6):3191
21. Wang H, Yu Q, Liu Y, Jin D, Li Y (2021) Spatio-temporal urban knowledge graph enabled mobility prediction. In: Proceedings
of the ACM on interactive, mobile, wearable and ubiquitous technologies

22. Sun Y, Li G, Du J, Ning B, Chen H (2022) A subgraph matching algorithm based on subgraph index for knowledge graph. Front Comput Sci 16(3):163606

23. Li Y, Liu J, Zhao H, Sun J, Zhao Y, Wang G (2021) Efficient continual cohesive subgraph search in large temporal graphs. World Wide Web 24(5):1483–1509

24. Willett P, Wilson T, Reddaway SF (1991) Atom-by-atom searching using massive parallelism: implementation of the ullmann subgraph isomorphism algorithm on the distributed array processor. J Chem Inf Comput Sci 31(2):225–233

25. Jin X, Lai L (2019) Mpmatch: a multi-core parallel subgraph matching algorithm. In: 2019 IEEE 35th international conference on data engineering workshops (ICDEW). IEEE, pp 241–248

26. Jin X, Yang Z, Lin X, Yang S, Qin L, Peng Y (2021) Fast: Fpga-based subgraph matching on massive graphs. In: 2021 IEEE 37th international conference on data engineering (ICDE)