基于CAN的地理语义数据存储与检索机制

卢海川 符海东 刘 宇

(武汉科技大学计算机科学与技术学院 武汉 430065)

(智能信息处理与实时工业系统湖北省重点实验室 武汉 430065)

摘 要 语义技术能够更智能、更精确地检索信息,辅助工作人员进行科学决策,已被应用于地理信息处理,并形成了基于RDF(Resource Description Framework)数据的地理查询语言GeoSPARQL。然而,基于地理语义信息处理的应用平台多采用中心化的存储和检索服务,使得这些平台存在单节点失效、扩展性差等缺陷。尽管已有研究人员提出了多种方法,试图利用对等网络技术来解决语义数据的分布式处理,从而提升应用系统的可靠性和扩展性,但这些方法并没有考虑地理语义数据自身的特征。针对上述问题,文中利用地理语义数据的特征在对等网络上对其进行存储,提出基于CAN(Content Addressable Network)的地理语义存储和检索方案,根据位置信息将地理语义数据映射到对等网络中,从而提高了语义数据的检索效率。实验结果表明,所提方案不仅具有良好的扩展性,而且地理信息的拓扑关系查询效率优于现有方案。

关键词 地理语义信息,RDF存储查询,GeoSPARQL,内容寻址网络

中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.02.027

Geo-semantic Data Storage and Retrieval Mechanism Based on CAN

LU Hai-chuan FU Hai-dong LIU Yu

(College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China)

(Hubei Province Key Laboratory of Intelligent Information Processing and Real Time Industrial System, Wuhan 430065, China)

Abstract Semantic technology can search information more intelligently and accurately, and assist researchers to make scientific decisions. Therefore, this technology has been introduced into geographic information processing and formed a geo-query language GeoSPARQL based on RDF (Resource Description Framework). However, the existing application platforms based on geographic semantic information processing adopt centralized storage and retrieval services, which will cause the disadvantages of single node failure and poor scalability. Although researchers have proposed a variety of methods to use peer-to-peer network to improve the reliability and scalability of application systems, these methods do not consider the characteristics of geographic semantic data. In view of the above problems, this paper considered the feature of geographical semantic data and optimized the storage of semantic data on the peer-to-peer network. This paper proposed a storage and retrieval scheme based on content addressed network, and also improved the retrieval efficiency of semantic data by mapping the triple to the network according to its position. The experimental results show that the proposed scheme has good expansibility, and the query efficiency of topology relation is superior to the existing schemes.

Keywords Geo-semantic information, RDF storage and query, GeoSPARQL, Content addressable network

1 引言

语义技术能够更智能、更精确地检索信息,辅助工作人员进行科学决策,该技术已经被应用于地理信息处理并形成了面向RDF数据的地理查询语言GeoSPARQL[1]。近年来,地理语义技术被广泛应用于灾难应急处理系统。地震、海啸等自然灾害发生时,受灾区域内的地理地貌被迅速改变,短时间内的地貌急剧变化会导致中心化服务存储的地理信息数据失去有效性。在灾难应急处理的过程中决策者需要根据最新的地理信息进行决策,最大程度地减少损失。基于对等网络的地理语义信息处理方案可以用于构建志愿者地理信息系统[2],节点在加入对等网络后可提供该区域最新的地理语义数据,帮助决策者根据实情做出准确判断,同时能够整合异构的地理数据,使用语义查询技术进行精确检索。基于对等网

到稿日期:2018-08-16 返修日期:2018-10-15 本文受国家自然科学基金(61673304,61272110,61502359),国家社会科学基金重大计划(11&ZD189)资助。

卢海川(1993-),男,硕士生,主要研究方向为语义网、分布式计算和信息检索,E-mail:lhch35@126.com;符海东(1971-),男,博士,教授,主要研究方向为软件工程、数据挖掘;刘 宇(1980-),男,博士,副教授,主要研究方向为语义网、自然语言处理和分布式计算,E-mail:liuyu@wust.edu.cn(通信作者)。

络的地理语义信息处理系统的另一个应用场景是车联网[3]。随着城市机动车数量的不断增加和交通需求的增长,交通拥堵问题越来越严重,运输效率低下。基于对等网络的地理语义信息处理系统的应用可以对车辆收集的地理信息进行语义描述,执行表达力更强的查询。

地理语义数据正在发挥越来越重要的作用,地理信息数据具有来源多样化、数据格式多样化的特点,将语义技术与地理数据结合能够发现信息的内部关联,充分利用多源信息可以体现数据的价值。然而,通过分析发现,现有的基于地理语义的信息处理应用平台多采用中心化的存储和检索服务,使得这些平台存在单节点失效、扩展性差等缺点。尽管已有研究人员提出了多种方法,试图利用对等网络技术解决语义数据的分布式处理,以提升应用系统的可靠性和扩展性,但这些方法并没有考虑地理语义数据的特征。本文提出了基于CAN[4]的地理语义数据处理方案,主要贡献如下:

1)提出了一种将地理对象相关三元组集合作为整体存储在对等网络中的存储结构,优化了数据检索效率。

2)提出了基于 CAN 的地理语义数据分布式存储方案,根据位置信息将地理对象映射到网络空间中,保留了地理对象的位置特性。

3)定义了地理语义拓扑查询类型,在分布式存储的基础上提出了高效的检索机制;对拓扑关系缓存、数据分布提出了优化方案。

2 相关工作

语义技术和地理信息技术的结合有利于整合多源异构的地理数据。通过用本体描述语言表示地理数据,能够增强数据的共享和复用,使得语义级别的地理信息检索表达力更强,能够更加精准地查询出符合语义条件的地理信息。诸多学者对地理语义存储和查询进行了研究。

Zhang 等于 2010 年提出了灾难应急处理管理中的地理特征自动搜索方案[5],指出了灾难应急处理中数据搜索的瓶颈不是缺少数据,而是无法智能搜索和无法整合异构的地理数据,并对此设计了易于使用者理解的查询接口,将使用者输入的查询语句解析为 SPARQL 查询语句进行查询,方便进行语义检索;同年,提出了基于 Web 的地理语义处理框架[6]。文献[7-8]提出使用机器学习的方法从 Web 文本中提取地理语义信息的方案,取得了良好的效果。为了将地理语义查询规范化,开放地理空间联盟于 2012 年提出 GeoSPARQL,定义了地理语义本体,并扩展了 SPARQL,使其支持地理语义查询,定义了空间关系谓词。诸多学者在地理语义查询优化方面也做了研究。段红伟等在分析了传统 RDF 数据组织方法和空间索引的基础上,提出了地理空间四元组模型,并基于该模型构建地理语义空间索引[9]。文献[10]提出使用潜在语义索引技术来增强地理语义查询的准确度,该方法具有较好的实践效果。Zhang 等于 2015 年提出了可互操作的在线志愿者地理信息系统[11],该系统通过机器学习的方式将地理数据与地理本体匹配进行统一存储。为了缓解 GeoSPARQL 查询缓慢的问题,文献[12]提出了一种基于并行化的处理方案,类似的方案还有文献[13]提出的方案,其使用 MapReduce[14]框架来提高地理语义查询效率。

同时,为了解决中心化 RDF 存储扩展性差、单节点失效的问题,研究人员在将 RDF 存储与对等网络相结合方面也做了诸多研究。Zhou 等提出了 S-RDF[15],将含有相同主语信息的三元组存储在相同节点上,通过判断语义相似度进行匹配查询。文献[16]在节点之间建立语义关系连接,通过语义连接进行查询。以上基于非结构化的方案由于网络本身的结构的限制,查询效率较低。文献[17-18]证明了类非结构化网络在大规模网络中不具备较好的扩展性。Cai 等基于 Chord[19]提出了 RDFPeers[20],该方法将 RDF 三元组进行三倍索引,对每一个要存储到网络中的三元组进行三倍冗余存储,查询效率较高。Monato 等提出了基于三维空间的 RDFCube[21],RDFCube 中的每个节点负责的区域是一个单元块,该方法将三元组视为一个三维的空间坐标,根据坐标进行存储。文献[22]提出了基于结构化对等网络支持分布式查询的 RDF 存储方案 CQRDFP2P,该方案能够处理多谓词的查询语句,将查询语句拆分为多条子语句,在查询过程中将查询结果动态合并。文献[23-24]对该方案建立了索引,并进行了查询优化。

现有的研究着重于地理语义信息处理和基于对等网络的语义数据分布式存储和查询。这些方案没有考虑地理语义数据的特殊性,地理对象的位置信息在计算拓扑关系等方面具有不可缺失的作用。本文在存储地理语义数据时进行了优化,保留了位置特性,使得在对等网络中可以高效地查询地理信息和拓扑关系。

3 基于 CAN 网络的地理语义存储

本节主要介绍基于 CAN 的地理语义数据存储方案。3.1 节介绍本体构建和存储方式;3.2 节描述网络中的节点结构;3.3 节介绍存储算法。

3.1 查询分类

本文实现了 GeoSPARQL 的部分本体定义,并参考主流的地理语义模型,使用单独的三元组表示地理对象的位置信息。本体的核心部分包括 SpatialObject 类和 Geometry 类。

SpatialObject 类为现实世界中的地理对象,如公园、湖泊、学校等;Geometry 类将地理对象映射为 3 类几何形状,分别为 Point(点)、Line(线)、Circle(圆)。SpatialObject 类通过属性 hasGeometry 与 Geometry 类连接。

为了便于描述地理对象的位置信息,本文扩展了如下谓词。

1) LocatedX:几何对象的横坐标或者经度,domain:Point,range:double;

2) LocatedY:几何对象的纵坐标或者纬度,domain:Point,range:double;

3)Center:几何对象中心点,domain:Line,Circle,range:LocatedX,LocatedY;

4)EndPointA:线对象的端点,domain:Line,range:Center;

5)EndPointB:线对象的端点,domain:Line,range:Center;

6)radius:半径,domain:Circle,range:非负实数。

本文将与地理对象相关的所有三元组集合作为整体,并

存储在网络中的同一个节点上。与地理对象相关的三元组集合不仅包含了所有与该地理对象相关的地理信息，还包含了与对象位置相关的三元组。本文对地理语义对象的三元组集合做出了如下定义。

定义 1（地理对象的三元组集合）　存在地理信息三元组集合 G，集合中的元素表示地理对象名称、位置、几何图形、区域等信息。对于某一地理对象 geo，存在三元组集合 O 包含一至多个三元组$\langle s,p,o\rangle$，所有的谓语 p 属于集合 G，并且主语是地理对象 S 或者为与地理对象 S 的位置信息相关的地理对象，则集合 O 为地理对象 geo 的地理语义对象。

例如地理对象 geo，其完整的地理语义对象包含以下三元组：

〈geo，geo：hasGeometry，Circle〉

〈geo，geo：Center，pointE〉

〈geo，geo：radius，60〉

〈pointE，geo：hasGeometry，Point〉

〈pointE，geo：LocatedX，60〉

〈pointE，geo：LocatedY，75〉

从上述描述中可以看出，地理语义对象不仅包含对象本身的信息，还包含与其关联的地理对象的信息。其中位置信息将用于地理语义对象在对等网络中的分布，根据不同对等网络的特性，将地理对象根据名称或者位置信息统一存储在同一个节点上。

根据 3 种地理对象位置信息的定义，可以将地理对象根据位置信息映射到对等网络中。同时将所有与地理对象相关的三元组集合作为整体，并将其存储在与其真实地理位置相关的网络节点中，以保存地理对象的地理位置信息。

3.2　节点结构

本文提出的分布式 RDF 存储系统由网络中的多个节点组成，所有节点自组成对等网络，每个节点都包含以下几个组成部分：底层网络、RDF 三元组加载器、RDF 数据库、Query 解析器和 Query 处理器，如图 1 所示。

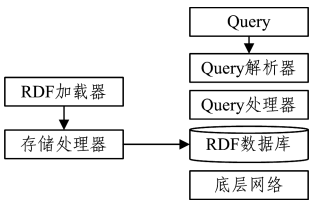


图 1　节点结构

Fig. 1　Structure of node

使用 CAN 作为底层网络；RDF 有多种描述语言，RDF 加载器对不同源或不同格式的 RDF 文件解析后，将结果传递给存储处理器；存储处理器用于处理地理对象。本文设计的方案中，地理语义数据源为包含完整地理语义信息的三元组集合，其中包含位置信息等与地理对象相关的多个三元组。存储处理器用于提取相同地理对象的所有三元组，形成地理语义对象三元组集合 O，将三元组集合 O 存储在对等网络节点的 RDF 数据库中；RDF 数据库可以是任何支持 SPARQL 查询语言的 RDF 数据库；Query 解析器用于解析原生的 SPARQL 查询语句，将其中的坐标、名称、阈值等信息提取出来，传递给 Query 处理器；Query 处理器将 Query 解析器解析出的有效信息整合于 RDF 数据库执行，并返回执行结果。

3.3　存储过程

基于 CAN 网络的地理语义数据存储方案的核心是将 RDF 三元组分布到网络中的各个节点，同时保留地理对象的位置信息。本文根据地理对象的真实坐标，将地理对象映射到网络中，保留了地理对象的物理分布，继而保存其地理相关性。

假设存在地理对象 S，其地理语义三元组集合为 O，集合 O 中存在完整的描述地理对象 S 的位置信息的三元组，则 O 为位置信息完整的地理语义对象。对于位置信息完整的地理语义对象，存储处理器首先提取出语义对象中的位置信息，直接使用位置信息生成的坐标(x,y)作为存储的目的地坐标，使用 CAN 的路由协议找到该目的地坐标，并将其存储在负责节点中。CAN 中的地理数据负责节点的定义如下。

定义 2（CAN 中的数据负责节点）　假设 CAN 中存在节点 p，节点 p 的负责区域为$[x_1,x_2,y_1,y_2]$，与地理对象 s 相关的三元组集合为 O，O 中存在$\langle s,\text{geo:LocatedX},x\rangle$和$\langle s,\text{geo:LocatedY},y\rangle$，如果 x 大于或等于 x_1 且小于 x_2，y 大于 y_1 且小于或等于 y_2，则 p 为该数据的负责节点。

地理语义对象分布方式既采用了结构化拓扑结构，在数据与节点之间建立了强联系，同时又根据真实地理坐标将对象映射到网络中，该方式保存了地理对象的地理相关性，保证了查询时可以根据对象的坐标进行拓扑关系计算，较为适合存储地理语义对象。

假设存在地理对象 S，其地理语义三元组集合为 O，集合 O 中缺少完整的描述地理对象 S 的位置信息的三元组，则 O 为位置信息缺失的地理语义对象。

对于位置信息缺失的地理语义对象，不能够根据坐标将数据映射到二维空间，无法保存其位置信息。本文采用一致性哈希算法，根据地理对象的名称生成一对虚拟坐标(x,y)，并将该坐标作为目的地坐标，采用相同的路由算法找到负责节点，将所有与地理对象相关的三元组存储在该节点。由于采用一致性哈希算法，在根据地理对象名称查询地理对象信息时，所产生的虚拟坐标是一致的，因此这种映射方式保证了系统可以根据地理对象名称快速定位到网络中的节点并查询出所有相关的信息。

地理语义三元组的存储处理过程如算法 1 所示。

算法 1　基于 CAN 的三元组定位算法

输入：需存储的三元组〈s，p，o〉，地理语义三元组集合 O，引导节点 bootstrap

输出：负责节点

```
Procedure locate_trilpe(〈s,p,o〉,O,bootstratp)
{
1. s=extract(〈s,p,o〉);//提取地理对象的名称信息即主语 s
2. 〈s_i,p_i,o_i〉=extract(s,O);//提取 O 中所有与 s 相关的三元组
3. (x,y)=extractLocation(〈s_i,p_i,o_i〉);//提取位置信息
4. if isComplete(x,y)=false
5.    (x,y)=createVirtualCoordinate(s);//根据名称生成虚拟坐标
6. end if
7. if (x,y) in bootstrap
8.    return bootstrap;
9. else peer=RouteToFind(x,y)
10.   return peer;
```

```
11. end if
}
```

由于真实世界中地理对象的分布可能是不均匀的，对于分布非常不均匀的数据会衍生出数据映射到网络后分布不均匀的问题，导致有的节点存储大量数据，而有的节点没有存储任何数据，从而造成负载不均衡。本文提出一种优化方案，该方案在系统路由找到目的地坐标所在节点后，遍历该节点的所有邻居，查询出每个节点 RDF 数据库所存储地理对象的数量，将地理对象存储在负载最低的节点，从而可以有效缓解数据分布不均匀的问题。本文实现了优化算法，并通过实验对其优化前后的性能进行了对比。

4 基于 CAN 网络的地理语义查询

由于将数据分布在网络中的各个节点上，查询时需要通过路由协议在节点间寻找目标节点，随着节点数量的增加，网络规模扩大时查询效率可能会降低。因此，使系统具有高扩展性和高查询效率是基于对等网络的地理语义处理方案的关键。本节详细介绍了如何在分布式地理语义存储的基础上进行高效查询。4.1 节介绍了查询分类，4.2 节－4.4 节介绍了每种查询的处理方法。

4.1 查询分类

本文将地理语义拓扑查询分为三大类：单对象地理信息查询、多对象拓扑关系确定、多对象拓扑范围查询。单对象地理信息查询表示查询与某一地理对象相关的所有地理信息，如湖泊的位置、大小等；多对象拓扑关系确定表示确定多个地理对象在空间上的拓扑关系，例如，北海与北海公园的拓扑关系为包含；多对象拓扑范围查询表示在一定范围内，查询与某一地理对象呈某种拓扑关系的所有地理对象，例如，查询 20 km 内与北海有相交关系的地理对象。

结合本文设计的分布式语义数据存储方案，以上 3 类查询可以细分为 6 类查询语句，表 1 描述了每种查询的查询条件、返回结果和查询语义。

表 1 地理语义数据的查询分类

Table 1 Query classification of geo-semantic data

序号	查询参数	返回结果	语义
Q1	geo:Coordinate(x,y)	地理对象所有相关信息	根据地理对象的坐标查询出地理对象所有的地理信息
Q2	geo:Name	地理对象所有相关信息	根据地理对象的名称查询出地理对象所有的地理信息
Q3	geo1:Name, geo2:Name	拓扑关系	提供对象 1 的名称和对象 2 的名称，确定它们的拓扑关系
Q4	geo1:Name, geo2:Coordinate(x,y)	拓扑关系	提供对象 1 的名称和对象 2 的坐标，确定它们的拓扑关系
Q5	geo1:Coordinate(x,y), geo2:Coordinate(x,y)	拓扑关系	提供对象 1 的坐标和对象 2 的坐标，确定它们的拓扑关系
Q6	geo:Name, geo:Coordinate(x,y), topological relation range	地理对象集合	提供地理对象的名称和坐标，指定查询范围，返回范围内所有与该对象呈某种拓扑关系的地理对象的集合

4.2 单对象地理信息查询

在网络中的引导节点接收到查询消息后，Query 解析器解析查询语句，判断通过坐标还是名称来查询地理对象，如果通过坐标查询，则提取出目的地坐标(x,y)。判断目的地坐标是否在当前节点的区域内，如果在，则查询当前节点 RDF 数据库，返回查询结果；否则，首先根据目的地坐标和当前节点中心计算目的地坐标位于当前节点的方向 *Direction*，然后分别计算 *Direction* 方向上所有邻居节点的中点到目的地坐标点的距离，获取距离最短的节点 p，将查询消息传递至节点 p。重复执行上述过程，直至找到负责节点，负责节点查询 RDF 数据库，并将查询结果返回。如果通过地理对象名称查询，则将上述过程的目的地坐标替换成根据名称生成的虚拟坐标，然后采用相同的策略进行查找。

可以根据地理对象的名称和坐标信息查询 CAN 中的单对象地理信息，这得益于在存储过程中，将地理对象的位置信息映射到了网络中，同时，对地理对象的名称采用哈希方法，生成一对虚拟坐标。路由算法为最短路径的贪婪算法，查询的平均跳数为$(d/4)(\sqrt[d]{n})$，其中 d 为空间维度，因此在二维空间下平均跳数为$\sqrt{n}/2$，该算法具有良好的扩展性，能够胜任节点数量的急剧增长。

4.3 多对象拓扑关系确定

多个空间对象的拓扑关系确定也是地理数据中较为常见的查询，本文基于 RCC-8[25] 模型判断空间拓扑关系。查询语句的参数可以为两个地理对象的对象名称，或一个对象名称和一组地理对象的坐标，或者两组坐标。

一个对象名称和一组坐标的查询语句的示例如下，该示例对应表 1 中的 Q4。

```
SELECT *
WHERE {
  circleA ?p ?s.
  ?s geo:LocatedX "37.5".
  ?s geo:LocatedY "37.5"
}
```

上述这个例子中，Query 解析器将提取出对象名称 *circleA* 和坐标(37.5,37.5)，由于只有一个明确的对象信息，无法直接利用 *circleA* 中的拓扑关系缓存判断拓扑关系，因此首先根据坐标路由查找坐标的负责节点，查询出该节点上的所有地理对象集合 G，遍历集合 G 中的地理对象，判断地理对象的拓扑关系缓存中是否缓存了 *circleA* 的拓扑关系，如果命中缓存，则将该地理对象和拓扑关系加入到结果集合中，仍通过 RCC-8 模型计算二者的拓扑关系，将对象和拓扑关系加入结果集。

查询单个地理对象的平均跳数为$\sqrt{n}/2$，查询两个地理对象的平均跳数为$\sqrt{n}$，因此本文方案在多对象的拓扑关系确定查询中也具有较高的效率。

4.4 多对象拓扑关系范围查询

多对象拓扑关系范围查询是指查找出一定范围内与某一地理对象呈某种拓扑关系的所有地理对象。然而在实际的查询中，用户往往并不关心所有对象，只关心一定范围的对象的拓扑关系，并且若想查询出所有符合关系的地理对象则需要查询网络中所有的节点，查询效率较低，用户往往希望能够通过限制查询跳数来快速获取查询结果。本文设计方案中的查询消息采用多路广播的机制，查询时将查询消息传递给当前节点的所有邻居节点，因此本文通过定义查询阈值来限制查

询时的查询范围，定义如下。

定义 3（查询阈值）　引导节点 *bootstrap* 接收多对象范围查询消息，路由查找到目标对象的负责节点，以负责节点为中心，消息向外传播的层数即为查询阈值。

为了评估拓扑关系范围查询的查询效率，本文定义了量化指标——拓扑查全度。

定义 4（拓扑查全度）　在多对象拓扑关系范围查询中，在最大查询跳数之内查询出的符合拓扑关系的地理对象集合为 G_A，网络中所有符合拓扑关系地理对象的集合为 G_B，两个集合数量的比例就是拓扑查全度。

多对象拓扑关系范围查询语句的示例如下，该示例对应表 1 中 Q6。

```
SELECT *
WHERE {
  threshold set "8".
  ?s1 geo:rcc8po ?s2.
  ?s1 geo:LocateX "53".
  ?s1 geo:LocateY "57".
  Filter(?s1=circleA)
}
```

treshold 为查询阈值，因此从查询语句中可以提取出查询阈值为 8。使用 Filter 关键字确定查询的地理对象 *circleA*，坐标为(53,57)，需范围查询的拓扑关系为 *PO*（相交）关系。首先从引导节点出发，查找出 *circleA* 的所有地理信息，然后开始执行多路广播机制，从引导节点开始，将查询消息并行传递给引导节点的所有邻居节点，在每个邻居节点中执行相同的查询语句，直至达到查询阈值。具体的实现过程如算法 2 所示。

算法 2　多对象拓扑范围查询算法

```
输入：查询语句 Q6，引导节点 bootstrap，查询层数 routeLayers
输出：查询结果集 Result
Procedure multi_objects_range_query(Q6,bootstrap,routeLayers)
{
1. queryObject=parse(Q6);//解析查询语句
2. name=queryObject.name;//获取名称
3. (x,y)=queryObject.coordinate;//获取位置信息
4. topo=queryObject.topo;//获取拓扑关系
5. threshold=queryObject.threshold;//获取查询阈值
6. geoObject=findInfo(name,(x,y));//获取地理语义信息
7. while routeLayers<=threshold //范围查询
8.    G=bootstrap.searchAllGeo();//获取节点上所有地理对象
9.    for all geo in G
10.      relation=computeRelation(geo,queryObject);//判断拓扑关系
11.      if relation=topo //符合拓扑关系则加入结果集
12.         Result.add(geo);
13.      end if
14.   end for
15.   routeLayers ++;
16.   multiBroadcast(bootstrap.neighbors);//多路广播查询
17. end while
}
```

从算法 2 中可以看出将地理对象映射到二维空间的优势，由于模拟了地理对象的真实分布，因此与某一地理对象具有一定拓扑关系的对象分布在其周围。该算法在网络中能够用尽量少的查询跳数查到更多的符合拓扑关系的地理对象。拓扑查全度随着查询阈值的增大而升高，并且增长速率较快，较为适合多对象拓扑关系范围查询。

5　实验设计与结果分析

本节将介绍实验框架和网络的模拟，实现本文中提出的存储和检索机制，并对实验结果进行分析。

5.1　实验框架

本文实验模拟实现了最多 2048 个节点数规模的多组网络。网络中的每个节点都有独立的空间，空间中所有的点由该节点负责。本文的模拟实现中，每个节点包含以下信息：1）节点 *ID*；2）节点边界坐标 x_1，x_2，y_1，y_2；3）邻居表。

我们同时模拟构建了相同节点数的对等网络 Gnutella[26] 和 Chord，分别实现了地理语义存储，并与本文的方案进行了比较。

实验数据集是由本文通过脚本模拟生成的 3000 个地理对象，为了使数据结构更紧凑、可读性更好，本文使用 Turtle 语法表示。为了方便拓扑关系的计算，以点类型地理对象和圆类型地理对象为实验对象。首先随机生成 1500 个点对象，位置信息 LocatedX 和 LocatedY 全部随机生成，范围为[0，100]，然后对应生成 1500 个圆对象。

5.2　实验设计与结果分析

本文实验的主要效率指标是查询路由跳数，而查询路由跳数主要取决于网络规模，因此本文进行了多组网络规模模拟实验，着重对比了不同规模网络下的查询跳数。为缓解复杂地理环境下 CAN 中节点数据分布不均匀的情况，我们提出了优化方案并对优化前后的性能进行了对比。

对于单对象地理信息查询，本文设置了 6 组实验，网络规模分别为 64，128，256，512，1024 和 2048 个节点。每组实验都随机选取 10 个地理对象进行查询，在基于 Gnutella 的方案、基于 Chord 的方案和本文方案中执行单对象地理信息查询，记录每一次的查询跳数，最终取 10 次查询跳数的平均值，结果如图 2 所示，其中横坐标和纵坐标均采用对数刻度。

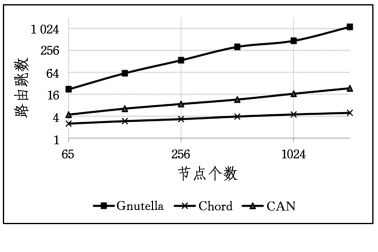


图 2　单对象地理信息查询效率的比较

Fig. 2　Comparison of efficiency of single object geographic information query

由图 2 可以看出，基于 Chord 的方案查询效率最高，平均跳数符合 $\log(N)$，随着网络规模扩大，查询跳数的增长速度极为缓慢，即使在 2048 个节点的网络中也只需要极少的跳数。基于 Gnutella 的方案查询效率最低，随着网络规模的增长，查询所需要的跳数急剧增长，对单对象的查询需要访问网络中近半数的节点。本文方案的查询效率介于 Gnutella 和 Chord 之间，平均查询跳数符合 $\sqrt{n}/2$，随着节点规模的增加，查询路由跳数增长缓慢，适用于大规模网络，具有良好的扩展性。

多对象拓扑关系确定查询采用了与精确匹配查询同样的分组，每组随机选取 10 对地理对象确定它们的拓扑关系，每组查询 10 次后取跳数平均值。图 3 为在 64，128，256，512，1024 和 2048 个节点规模网络下的查询平均跳数。其中的坐标为对数刻度。从图 3 中可以看出，3 种网络下的平均跳数增长趋势与单对象地理信息查询大致相同，由于要查询两个地理对象信息，因此平均查询跳数基本都为单对象查询的两倍。基于 Chord 的方案优势更加明显，即使在 2048 个节点的网络中，10 跳以内就可以确定拓扑关系。本文方案在效率上虽然不如基于 Chord 的方案，但效率也较高。基于 Gnutella 的方案的两次查询跳数之和接近网络中的节点数量，不具备良好的扩展性。

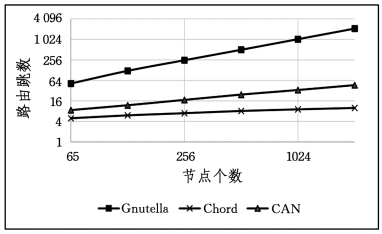
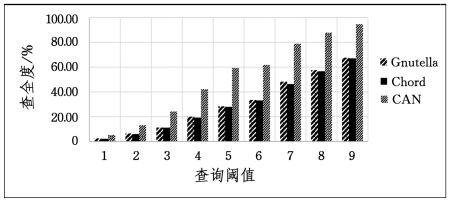
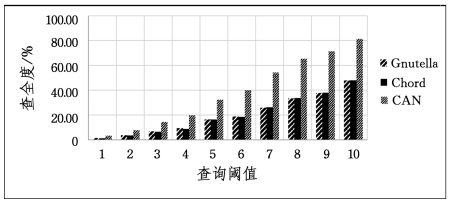


图 3 多对象拓扑关系确定查询效率的比较

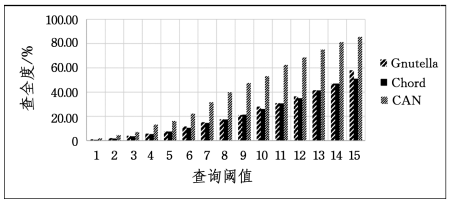
Fig. 3 Comparison of efficiency of multiple objects topological relation match



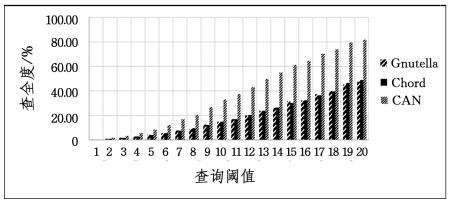
(a)256 个节点



(b)512 个节点



(c)1024 个节点



(d)2048 个节点

图 4 多对象拓扑关系范围查询查全度比较

Fig. 4 Comparison of completeness of multiple objects topological range query

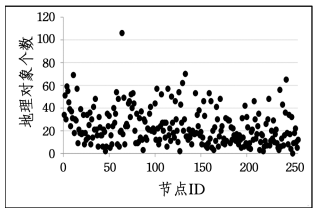
对于多对象拓扑关系范围查询，本文设置 4 组对照实验，分别在节点规模为 256，512，1024 和 2048 节点数网络中。由于本文提出的基于 CAN 的多对象拓扑关系范围查询方案中需要设定查询阈值，而在 Gnutella 和 Chord 中没有与之对应的量化指标，为了使实验结果更明显，以本文方案中设定不同的阈值后得出的查询跳数作为 Gnutella 和 Chord 的最大查询跳数，统计在最大查询跳数限制下查询出的符合拓扑关系的地理对象，对比拓扑关系查全度。图 4 为 3 种网络在 256，512，1024 和 2048 个节点规模网络下的拓扑关系查询度对比。

从图 4 中可以看出，本文设计的基于 CAN 的方案明显优于基于 Gnutella 的方案和基于 Chord 的方案。图 4(a)所示的 256 个节点的网络中，当查询阈值为 9 时，本文系统的查询跳数为 140～200 左右，在相同跳数下，本文方案的拓扑关系查全度为 94.46%，基于 Gnutella 方案的拓扑关系查全度为 67.51%，基于 Chord 方案的拓扑关系查全度为 66.9%，这意味着本文方案可以用尽量少的查询跳数，查询出尽量多的符合拓扑关系的地理对象，较为适合多对象拓扑关系范围查询。这是由于与查询地理对象有一定拓扑关系的对象都分布在该对象的周围，从负责节点出发向外“扩散”查找可以有效避免无效的查询，从而找到更多符合拓扑关系的地理对象。

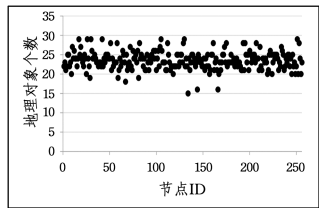
在 256 个节点的网络中，本文系统在查询阈值为 9 时就查询出了几乎所有的地理对象，在 512 个节点的网络中查询阈值为 10 时，查全度约为 80%，在 1024 个节点的网络中查询阈值为 15 时，查询出接近 85%的地理对象，在 2048 个节点网络中查询阈值为 20 时，查询出 81%的地理对象。这是因为随着网络规模的扩大，地理对象分布得越来越分散，路由节点数增多，导致需要设定更高的查询阈值才能保持较高的查全度。

5.3 数据分布优化

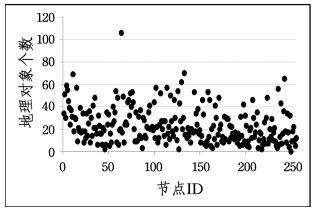
本文第 2 节提出了地理对象分布不均匀的情况会造成存储到网络后节点负载不均衡的问题，对此本文根据 CAN 的特性提出了优化方案。在路由找到目的地坐标点所在的节点时，查询该节点的所有邻居节点存储的地理对象数量，将数据分布在负载最小的节点上。本文使用脚本生成 6000 个模拟地理对象，在 256 个节点和 512 个节点的网络中进行对比实验。



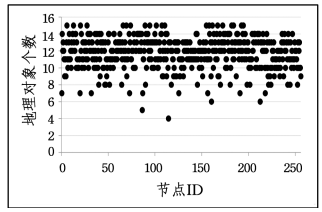
(a)256 个节点(优化前)



(b)256 个节点(优化后)



(c)512 个节点(优化前)



(d)512 个节点(优化后)

图 5 节点负载优化前后的数据分布对比

Fig. 5 Data distribution comparison before and after node load optimization

从图 5 可以看出，优化后所有节点地理对象分布在 15～30 个之间；512 个节点的网络中优化前最多存储地理对象 59 个，24 个节点没有存储任何地理对象，优化后每个节点上的地理对象大约为 8～15 个。这证明了优化方案在数据集分布不均匀的情况下能够有效分散数据存储，均衡网络负载。

结束语　本文提出的基于 CAN 的地理语义存储和检索方案对地理语义数据进行了优化，具有良好的扩展性，在大规模网络中也保持着较高的查询效率。特别是多对象拓扑关系范围查询能够极大地发挥网络拓扑优势，但仍有优化的空间。针对如何解决网络中节点失效导致的数据丢失问题，我们将重点研究新节点加入和节点退出时语义数据的迁移和网络结构的重新组织，以保证网络的健壮性。同时，我们将尝试对 CAN 的路由算法进行优化并引入语义推理来帮助判断拓扑关系。

参 考 文 献

[1] BATTLE R, KOLAS D. Enabling the geospatial semantic web with parliament and geosparql[J]. Semantic Web, 2012, 3(4): 355-370.

[2] BALLATORE A, WILSON D C, BERTOLOTTO M. A survey of volunteered open geo-knowledge bases in the semantic web [M]// Berlin: Quality Issues in the Management of Web Information. Springer, 2013: 93-120.

[3] GERLA M, LEE E K, PAU G, et al. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds[C]// 2014 IEEE World Forum on Internet of Things (WF-IoT). IEEE, 2014: 241-246.

[4] RATNASAMY S, FRANCIS P, HANDLEY M, et al. A scalable content-addressable network[C]// Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. ACM, 2001: 161-172.

[5] ZHANG C, ZHAO T, LI W. Automatic search of geospatial features for disaster and emergency management[J]. International Journal of Applied Earth Observation and Geoinformation, 2010, 12(6): 409-418.

[6] ZHANG C, ZHAO T, LI W. The framework of a geospatial semantic web-based spatial decision support system for digital earth[J]. International Journal of Digital Earth, 2010, 3(2): 111-134.

[7] CRUZ I F, GANESH V R, MIRREZAEI S I. Semantic extraction of geographic data from web tables for big data integration [C]// Proceedings of the 7th Workshop on Geographic Information Retrieval. ACM, 2013: 19-26.

[8] YU L, LU F, ZHANG X, et al. Context Enhanced Keyword Extraction for Sparse Geo-Entity Relation from Web Texts[C]// Asia-Pacific Web Conference. Springer, Cham, 2016: 253-264.

[9] DUAN H W, MENG L K, HUANG C Q, et al. A Method for Geo Semantic Spatial Index on SPARQL Query[J]. Acta Geodaetica et Cartographica Sinica, 2014, 43(2): 193-199. (in Chinese)
段红伟，孟令奎，黄长青，等. 面向 SPARQL 查询的地理语义空间索引构建方法[J]. 测绘学报，2014，43(2)：193-199.

[10] LI W, GOODCHILD M F, RASKIN R. Towards geospatial semantic search: exploiting latent semantic relations in geospatial data[J]. International Journal of Digital Earth, 2014, 7(1): 17-37.

[11] ZHANG C, ZHAO T, LI W. Towards an interoperable online volunteered geographic information system for disaster response [J]. Journal of Spatial Science, 2015, 60(2): 257-275.

[12] ZHAO T, ZHANG C, ANSELIN L, et al. A parallel approach for improving Geo-SPARQL query performance[J]. International Journal of Digital Earth, 2015, 8(5): 383-402.

[13] ZHANG C, ZHAO T, ANSELIN L, et al. A Map-Reduce based parallel approach for improving query performance in a geospatial semantic web for disaster response[J]. Earth Science Informatics, 2015, 8(3): 499-509.

[14] DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113.

[15] ZHOU J, HALL W, DE R D. Building a distributed infrastructure for scalable triple stores[J]. Journal of Computer Science and Technology, 2009, 24(3): 447-462.

[16] GIUNCHIGLIA F, KHARKEVICH U, HUME A, et al. Semantic flooding: search over semantic links[C]// 2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW). IEEE, 2010: 191-196.

[17] SAROIU S, GUMMADI P K, GRIBBLE S D. Measurement study of peer-to-peer file sharing systems[C]// Multimedia Computing and Networking 2002. International Society for Optics and Photonics, 2001, 4673: 156-171.

[18] SEN S, WANG J. Analyzing peer-to-peer traffic across large networks[C]// Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement. ACM, 2002: 137-150.

[19] STOICA I, MORRIS R, LIBEN-N D, et al. Chord: a scalable peer-to-peer lookup protocol for internet applications[J]. IEEE/ACM Transactions on Networking (TON), 2003, 11(1): 17-32.

[20] CAI M, FRANK M, YAN B, et al. A subscribable peer-to-peer RDF repository for distributed metadata management[J]. Web Semantics: Science, Services and Agents on the World Wide Web, 2004, 2(2): 109-130.

[21] MATONO A, PAHLEVI S M, KOJIMA I. RDFCube: A P2P-based three-dimensional index for structural joins on distributed triple stores[M]// Databases, Information Systems, and Peer-to-Peer Computing. Springer Berlin Heidelberg, 2007: 323-330.

[22] LIAROU E, IDREOS S, KOUBAEAKIS M. Publish/subscribe with RDF data over large structured overlay networks[M]// Databases, Information Systems, and Peer-to-Peer Computing. Springer Berlin Heidelberg, 2007: 135-146.

[23] PELLEGRINO L, HUET F, BAUDE F, et al. A distributed publish/subscribe system for RDF data[C]// International Conference on Data Management in Cloud, Grid and P2P Systems. Springer Berlin Heidelberg, 2013: 39-50.

[24] ALI L, JANSON T, SCHINDELHAUER C. Towards Load Balancing and Parallelizing of RDF Query Processing in P2P Based Distributed RDF Data Stores[C]// Euromicro International Conference on Parallel. 2014.

[25] NIKOLAOU C, KOUBARAKIS M. Fast consistency checking of very large real-world RCC-8 constraint networks using graph partitioning[C]// Twenty-eighth Aaai Conference on Artificial Intelligence. 2014.

[26] RIPEANU M. Peer-to-peer architecture case study: Gnutella network[C]// Proceedings First International Conference on Peer-to-Peer Computing. 2001: 99-100.