# 基于 SPARQL 的 RDF 数据节点间关系路径检索

肖竹军

(武汉理工大学 计算机科学与技术学院,湖北 武汉 430063)

**摘 要**：在分析 SPARQL 标准和基于 Jena 的开源 SPARQL 工具 ARQ 查询引擎源码的基础上,提出了可支持关联查询的扩展 SPARQL 标准及其设计和实现方案,认真分析了已有的试验成果。

**关键词**：SPARQL;ARQ;Jena;RDF;关系路径;关系查询

**中图分类号**：TP391 **文献标识码**：A **文章编号**：1674-7720(2011)09-0050-04

# Relationship path search between RDF data nodes based on SPARQL

Xiao Zhujun

(School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, China)

**Abstract**: A careful analysis of the SPARQL standard and the source code of ARQ query engine based on Jena has been carried out in the paper. A solution to extend the SPARQL standard is proposed and implemented, and the experimental results are analysed carefully at last.

**Key words**: SPARQL; ARQ; Jena; RDF; relationship path; relational query

资源描述框架 RDF(Resource Description Framework)[1]是 W3C 组织基于可扩展标记语言(XML)开发的一种元数据描述框架。RDF 能够定义概念以及概念间的关系,描述易被机器理解的信息和知识。它提供的语义模型可用于描述 Web 上的任意资源及其类型,为网上资源描述提供了一种通用表示框架,解决语义异构问题,实现数据集成的元数据解决方案。同时,RDF 可用于数据发现,为搜索引擎提供更强大的搜索功能。RDF 通过基于 XML 语法明确定义的结构化约定来建立语义协定与语法编码之间的桥梁,以此促进元数据的互操作能力。因此,RDF 是解决计算机知识表示问题的最佳选择,可以很好地描述元数据。

SPARQL (Simple Protocol and RDF Query Language)是为 RDF 开发的一种查询语言和数据获取协议,虽然它是为 W3C 开发的 RDF 数据模型定义,但是可以用于查询任何可以用 RDF 来表示的信息资源。现行 SPARQL 能够满足类似 SQL 中基本模式匹配、分组、连接、合并等查询形式,并能够根据用户定义有效地返回映射结果集,能够满足基于 RDF 数据的基本查询需求。

RDF 数据的精髓在于以半结构化数据形式来存储知识以及知识间的基本关系,比较遗憾的是,目前的 SPARQL 标准及工具还没有提供一种有效的途径来查询任意指定的两节点间可能存在的各种关系路径以及任意指定节点周围可能散射的各种关系路径。为了解决如上问题,本文在原有的 SPARQL 标准的基础上引入新关键词来描述关联查询语义,并在针对基于 Jena[2]的SPARQL 开源引擎 ARQ 基础上进行实验、支持扩展新的标准及功能的方案。

## 1 SPARQL 语法及扩展

### 1.1 SPARQL 基本语法

#### 1.1.1 基本术语

被“<>”界定的术语是 IRI 参考[RFC3987],它们代表 IRIs,直接地或相对于一个基本的 IRI。IRIs 是 URIs [RFC3986] 的一般化,而且完全与 URIs 和 URLs 兼容。SPARQL 为 IRIs 提供两种缩写机制:namespace 前缀和相关的 URIs。查询术语可能是文字字符串(用双引号“”或单引号‘’括起来),有一个可选择的语言标签或可选择的数据类型 IRI。为方便起见,整数能被直接地写,且被解释成 datatype 的类型文字 xsd:integer;十进制数被解释为 xsd:decimal,含有一个指数的数被解释为 xsd:double。类型值 xsd:boolean 也能被写为 true 或 false。SPARQL 查询变量有全局范围,查询时,同一个名字在各

处是相同的变量，变量用“？”指出[3-4]。

1.1.2 三元组模式

三元组模式被写作为一列主语、谓语和宾语，用简单的方法来写一些通用的三元组模式，用{}将其聚集在一起。

1.1.3 图模式

SPARQL 查询语言是基于图模式匹配的，最简单的图模式是三元组模式，如同一个 RDF 三元组，但在任何主语、谓语或宾语的位置中可能有变量，如{？Book dc：title？title}。复杂图模式能由简单图模式组合而成，常见的复杂图模式是基本图模式、组合图模式、可选择图模式、联合图模式、RDF 数据集图模式、值约束条件六种模式中的一种。

1.1.4 值约束

SPARQL 中的 FILTER 关键字对绑定变量的值进行约束，从而限制查询的结果。这些值约束条件是对布尔值进行计算的逻辑表达式，并且可以与逻辑操作符 && 和||组合使用。例如，可以用过滤器把返回名称列表的查询修改为只返回和指定正则表达式匹配的名称。

1.1.5 查询类型

SPARQL 支持 SELCET、ASK、DESCRIBE 和 CONSTRUCT 四种类型的查询。典型的 SPARQL 查询由 SELCET、FROM、WHERE 三部分组成。SELCET 子句指定查询应当返回的内容；FROM 是一个可选的子句，提供了将要使用的数据集的 URI，可以指向一个本地文件，也可以指向 Web 其他地方的某一个图的 URL；WHERE 子句由一组三元模式组成，采用基于 Turtle 的语法表示。这些三元模式共同构成了所谓的图形模式。ASK：应用程序可以使用 ASK 形式来测试查询模式是否有一个解决方案。如果查询的图形模式在数据集中有匹配物，那么 ASK 将返回“yes”；如果没有匹配物，则返回“no”。DESCRIBE：返回一个图形，其中包含和图形模式匹配的节点的相关信息。例如，DESCRIBE？person WHERE{？person foaf：name“Jon Foobar”}会返回一个图，其中包括来自 JonFoobar 的模型的三元模式。CONSTRUCT：用来为每个查询结果输出一个图形模式，这样就可以直接从查询结果创建新的 RDF 图。

1.2 SPARQL 语法扩展

扩展的基本原则是不影响原有 SPARQL 查询语法及特性，通过引入适当的查询关键字来描述查找指定节点间的关键关系。第一个关键词为 SEEK，用来表示路径查询类型，其用法类似于 SELECT、ASK、DESCRIBE、CONSTRUCT 等关键字；第二个关键词为 START，用来描述关系路径的起点图模式；第三个关键词为 END，用来表示关系路径的终点图模式；第四个关键词为 NODE，用来描述关系路径节点的图模式，以及与起点图模式和终点图模式间的连接方式；第五个关键词为 CONSTRAINT，用来描述关系路径的约束，作用类似于 FILTER 关键词，主要用来限制关系路径最短长度、关系路径最长长度、与起点和终点的关系变量前缀、路径节点变量前缀等，其中最短路径为默认值为 3，最长路径长度默认值为 6。遵循以上扩展标准的节点间关系路径查询语法如下。

```
SEEK ？node，？power，？link
WHERE {
    START{
        ？s1 <http://jena.test/elements/type> “country”；
        ？s1 <http://jena.test/elements/name> “A 国”.
    }
    END{
        ？s2 <http://jena.test/elements/type> “country”.
        ？s2 <http://jena.test/elements/name> “B 国”.
    }
    NODE{
        ？s1 ？link ？node.
        ？node ？link ？s2.
        ？node<http://jena.test/elements/type>“keypoint”.
        ？node <http://jena.test/elements/power>？power.
        Filter(？power>1)
    }
    CONSTRAIT{
        LinkName(“link”)
        NodeName(“node”)
        MinDepth(3)
        MaxDepth(6)
    }
}
```

当只有 START 描述的起点而没有 END 描述的终点时，则用来查找从该起点出发的符合路径节点中描述的关系路径，CONSTRAIT 依然用来描述路径的各种约束信息、含义及用法不变，到此便基本完成了对 SPARQL 标准及语法的扩展工作。

## 2 基于 ARQ 开源引擎的扩展

2.1 创建查询语法树

ARQ[5]目前支持标准 SPARQL 语法树的生成，在扩展标准之后需要加入新的语法树创建规则，以识别扩展标准中新加入的关键词，最终建立正确的语法树结构[6-7]。新规则的引入需要保证在 SEEK 查询中至少在 WHERE 子句中出现 START 关键字来描述关系路径起点的模式，而 NODE 关键字来描述关系路径中间各节点的模式，最后用 CONSTRAIT 关键字来限制关系路径长度等属性。其中 END 关键字为可选路径，如果没有关系终点限制，则只需要达到关系路径最大深度后停止关系查询即可。若不能满足上述基本规则，则查询语法树创建失败，可向客户端反馈语法错误。扩展后的语法树基本逻

辑结构如下所示。另外,查询语法树对象只是一个中间形式,主要目的是为了生成查询代数表达式服务。

```
(project(? node , ? link , power)
 (path(
      start(
           bgp(
                triiple(? s1 <http:
                      //jena.test/elements/type> "country")
                triiple(? s1 <http:
                      //jena.test/elements/name> "A 国")
            )
      )
  end(
      bgp(
           triiple(? s2 <http:
                      //jena.test/elements/type> "country")
                triiple(? s2 <http:
                      //jena.test/elements/name> "B 国")
           )
  )
  node(
      bgp(
      triiple(? s1 ? link ? node)
           triiple(? node ? link ? s2)
           triiple(? node<http:
                    //jena.test/elements/type> "keypoint")
           triiple(? node <http:
                    //jena.test/elements/power> ? power)
           )
      )
 ))))
```

### 2.2 生成查询代数表达式

ARQ 引擎在正确生成查询语法树后则通过查询语法检查,此时可将语法树中描述的操作转换为 ARQ 引擎中与之对应的算术运算操作或者算术运算操作组合[8]。在 ARQ 引擎中,每种算术运算操作都有一个与之对应的运算操作类型对象,用来封装和存储该运算的基本信息以及与其他运算操作之间的协作关系,以达到最终的查询目的。

扩展 SPARQL 标准后,ARQ 引擎中并不存在与扩展后的关键词对应的算术运算操作类型,即无法将语法树正确地转换为查询代数表达式。因此,在 ARQ 引擎中引入新的算术操作对象类型(PathOp),该操作类型需要组合起点运算和终点运算两个基本模式运算对象(BgpOp)以及自身的路径探索连接运算。PathOp 的基本设计原则是在重用已有运算的基础上加入必要的运算操作达到路径探索连接的目的,保证原有引擎设计结构与实现方法。

### 2.3 查询优化与运算

成功生成查询代数表达式后,就可以针对查询代数表达式进行优化。绝大部分查询代数运算同样遵循代数运算中的结合定律与分配定律,进行合理排列组合达到降低时间和空间复杂度的目的。新的查询运算建立在原有运算基础之上,可以保持针对原有运算的优化原则。

最后是对运算表达式进行运算,ARQ 引擎采用在算术运算部分使用修饰者模式,具体运用了结果流迭代器技术,为新的扩展留下空间,同时也尽量保证运算的时空效率。因此在实现扩展运算的时候只需继承或封装上级结果流迭代器,在本迭代器中实现路径探索连接的运算过程。目前采用向后迭代、向前迭代和双向迭代三种迭代算法。

(1)向后迭代。首先将起点放入候选队列,取出候选队列第一项与数据集中的其他节点进行迭代匹配,若匹配成功并且匹配深度小于最大深度约束,则将本条匹配加入候选队列;若与终点连接成功,则该匹配为一条匹配路径并返回。重复上述过程,直到候选队列为空。

(2)向前迭代的过程与向后迭代完全相同,只是将终点放入候选队列,然后开始迭代过程。

(3)双向迭代。让向前与向后迭代同时进行,每个方向都有一个候选队列,每次向前队列匹配成功则与向后候选队列中的项进行匹配连接,若连接成功,则返回该连接匹配,反之亦然。双方将各自匹配成功的并且小于最大深度约束的匹配项加到各自的候选队列,当其中任意一方的候选队列为空时,迭代结束。

## 3 实验结果分析

### 3.1 节点间关系路径

节点间关系路径实验在测试数据集上进行,查询条件与图 1 基本一致。该实验主要测试查询节点间可能存在的关系路径,选取的起点为"A 国",终点为"B 国",查询出来的路径较多,经过 power 因子过滤和修正之后,主要有 3 条路径,图 1 给出了这 3 条关系路径的基本逻辑关系图。

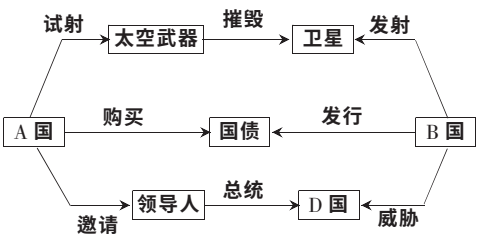


图 1 节点间关系路径

### 3.2 单节点周围散射关系路径

单节点周围散射关系路径实验主要测试在不限定终点的情况下查询单节点周围散射的各种关系路径。本实验仅限制起点"A 国",同样查出的路径比较多,经过 power 因子过滤和修正后主要有 8 条路径,图 2 给出了这 8 条路径的基本逻辑关系图。

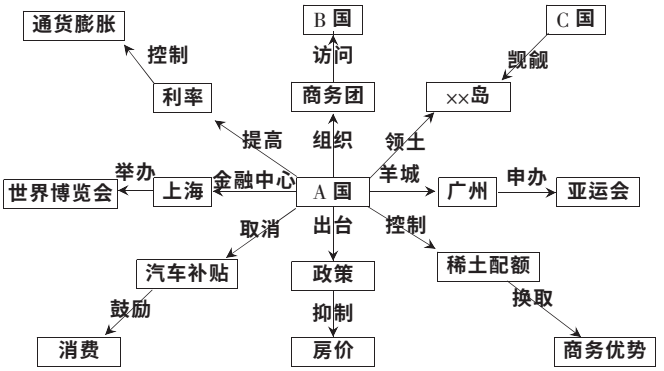


图 2 单节点散射关系路径

这两类实验测试表明，所做扩展基本能达到预期的目标，但同时也存在一定的问题。在比较复杂和宽泛的语义环境下，节点间的关系很多而且比较繁杂，如何从如此多的关系路径中搜寻到有意义的路径仍然是一个难题。本实验尝试使用一个 power 因子来描述单个关系的重要性，在一定程度上可以缓解这个问题，但仍然还有较大的不足之处。

本文所要解决的问题是在项目实践中提出的，目的是为了丰富和完善现有的 SPARQL 标准及功能，增强 RDF 数据的实际应用能力，满足语义网应用与开发中对关系路径搜索能力的需求。通过阅读大量的文档及文献，提出了比较合理的 SPARQL 扩展标准，并认真阅读开源 ARQ 查询引擎源码，在原有功能的基础上植入新的关系路径搜索功能，相信会进一步提升 SPARQL 及 RDF 的应用前景。下一步的工作计划是进一步完善和加强 SPARQL 在关系搜索领域的能力。例如，引入排序因子等技术提升查找路径的关键性与有效性，同时进一步提高 SPARQL 搜索性能。

## 参考文献

[1] Resource Description Framework (RDF)[EB/OL]. http://www.w3.org/2001/sw/wiki/RDF,2011-01-03.

[2] Jena-a semantic Web framework for Java [EB/OL]. http://openjena.org/index.html,2011-01-03.

[3] Sparql2sql a query engine for SPARQL over jena triple stores [EB/OL]. http://jena.sourceforge.net/sparql2sql/,2007-11-10.

[4] SPARQL Query Language for RDF [EB/OL]. http://www.w3.org/TR/rdf-sparql-query/,2008-01-15.

[5] ARQ-SPARQL Processor for Jena [EB/OL]. http://openjena.org/ARQ/,2011-01-03.

[6] Extensions in ARQ [EB/OL]. http://openjena.org/ARQ/extension.html,2011-01-03.

[7] ARQ-Extending Query Execution[EB/OL]. http://openjena.org/ARQ/arq-query-eval.html,2011-01-03.

[8] ARQ-SPARQL Algebra [EB/OL]. http://openjena.org/ARQ/algebra.html,2011-01-03.

（收稿日期：2011-01-10）

作者简介：

肖竹军，男，1985 年生，硕士研究生，主要研究方向：软件工程。