一种简历语义搜索系统的实现方法

柯叶青[1,2] 马志柔[2] 伍海江[1,2] 刘 杰[2]
(中国科学院大学计算机与控制学院 北京 100190)[1] (中国科学院软件研究所 北京 100190)[2]

摘 要 政府与企事业单位的人事部门通常面临着如何从大量简历中筛选出合适人选的问题。一些业务部门对人才的需求通常只能表达为标签,比如"具有丰富搜索引擎开发经验"、"985高校毕业"等,这些需求不能通过SQL查询或关键词搜索来实现。为了解决这个问题,提出一种基于语义网的简历搜索方法。通过领域知识库辅助信息提取的方法,对简历信息进行语义分析和标签的自动生成。此外,在大规模人员情况下,提出了多层次缓存的方法,极大提高了性能。将该方法应用于某机构一万人员简历的筛选,实验结果表明了其有效性。

关键词 语义网,RDF,SPARQL,性能优化
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.12.013

SmartHR: A Resume Query and Management System Based on Semantic Web

KE Ye-qing[1,2] MA Zhi-rou[2] WU Hai-jiang[1,2] LIU Jie[2]
(School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100190, China)[1]
(Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)[2]

Abstract The personnel departments of organizations are always confronted with the challenge of efficiently finding out suitable candidates from massive resumes. Some business departments usually express their demand for talents as tags, such as "with rich search engine development experience", "graduated from 985 universities" and so on. These requirements cannot be done by SQL queries or keyword search. To fill this gap, this paper proposed a resume analysis and search method based on semantic Web. By the method of domain knowledge base auxiliary information extraction, resume information is semantically analyzed and labels for suitable candidates are automatically generated. In addition, on a large personnel scale, we proposed a method of multi-level cache by which the performance has been greatly improved. Meanwhile, it is applied to nearly ten thousand personnel agency resumes and experiments show the effectiveness.

Keywords Semantic Web, RDF, SPARQL, Performance optimization

1 综述

现在各种组织机构面临着一个巨大挑战,即如何从大量的半结构化简历中精准高效地筛选到合适人选。依赖于人工筛选的方式不仅耗时,而且具有很强主观性。有些机构采用SQL查询或关键词匹配的方法,却不适用于很多场景。例如,某IT公司想找到一个"具有alexa排名前100的公司工作经验"的人,而公司里的简历信息只包含像谷歌、facebook、优酷、百度等这种工作单位的信息,不直接包含"alexa排名前100"关键词,并且alexa排名又是动态更新的。

为了解决这个问题,本文提出了一种基于语义网[1]的简历分析与搜索方法。它借助于领域知识库,能够很好地理解用户的意图;同时能从简历信息中抽取到一些领域知识,进而对简历信息进行语义分析,生成相应人员标签,有助于更加精准地搜索到合适人选。语义网因其能够提供一种通用框架,允许数据在多个应用和企业之间共享复用,现已得到了广泛的使用。在本文中,异构的语义信息采用RDF数据模型表示,并使用SPARQL[2]查询。鉴于这些优点,该方法不仅能够提供一种可扩展、可复用的知识管理方式,还能为用户提供简单的语义搜索接口,即使不熟悉SPARQL或者不理解本体等相关概念,也能搜索到想要的结果。

本文的主要贡献在于:1)通过领域知识库辅助信息提取的方法,对简历信息进行语义分析和标签自动生成;2)在大规模人员的情况下,提出了多层次缓存方法,实现性能的极大提高。本文第2节介绍系统的总体思路和架构;第3节详细介绍知识库的构建和标签的提取;第4节介绍性能优化;第5节是实验;第6节给出相关工作;最后总结全文。

2 系统总体思路及架构

本节介绍系统的总体思路及架构。如图1所示,该架构主要由4个部分组成:关系型数据库到RDF数据存储的转换模块、知识库引擎模块、知识库管理模块和前端用户交互模

到稿日期:2015-03-02 返修日期:2015-06-30 本文受国家自然科学基金项目(61170074,61202065),国家高技术研究发展计划(2012AA011204)资助。

柯叶青(1989-),女,硕士生,主要研究方向为语义搜索、大数据分析,E-mail:keyeqing12@otcaix.iscas.ac.cn;马志柔(1980-),男,硕士生,助理研究员,主要研究方向为网络分布式计算与软件工程;伍海江(1986-),男,博士生,主要研究方向为人工智能与知识工程;刘 杰(1982-),男,博士,副研究员,主要研究方向为大数据分析处理、分布式系统。

块。这些模块之间是松耦合的，它们可以被其它的应用单独采用。最左边的是数据映射模块，它可以从基于关系型数据库存储的简历系统中提取到半结构化的数据，并将其作为输入，映射到 RDF 数据模型。前端用户交互模块可以使用户和服务器更加友好地进行交互。知识库管理和知识库引擎模块是系统的核心。它们依赖于动态知识库管理策略，其中包含了大量满足用户不同需求的功能模块。在知识库引擎模块中，Java API 作为底层查询引擎和上层知识库管理层的中间层，具有灵活、可扩展的结构。本文将着重介绍两个关键点，即知识库构建和标签的自动生成以及性能优化，来揭示语义网应用的技术细节与挑战。

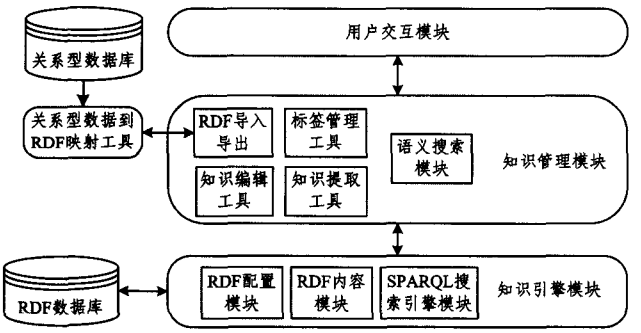

图 1　系统整体架构

3　知识库构建和标签的自动生成

知识库的构建和人员标签的提取是语义分析搜索的基础。本文通过图 2 所示的 4 种方式构建知识库。首先，将遗留下来的关系型数据库转换成 XML 文件，然后将 XML 文件映射成 RDF 文件，最后将 RDF 文件导入到 RDF 数据存储系统中。另外，本文还通过批量导入、知识提取算法维护领域知识库。构建好了基本的领域知识库后，就可以对简历信息进行语义分析，进一步给人员生成不同的标签。

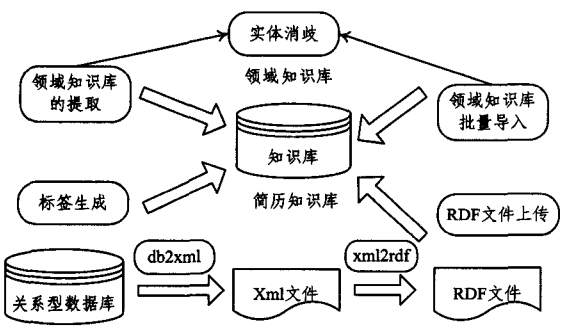

图 2　知识库构建

3.1　知识库的构建

3.1.1　知识库的本体设计

本体设计主要包括对概念、关系以及实体的设计，如图 3 所示。概念是指一类实体的集合，本文具体指候选人概念、简历信息概念和一些领域知识库。关系是指整体和部分的关系以及从属关系。整体和部分的关系被分为数据属性和对象属性两类。对象属性是指实体和实体之间的关系，数据属性是指实体和文本之间的关系。例如，候选人的毕业院校是候选人的对象属性，而候选人的名字没有特殊的意义，只是数据属性。从属关系包括概念和实体之间的关系以及子概念和父概念之间的关系。实体是概念中的一个个体。例如，候选人概念中的每个人都是该概念的实体。

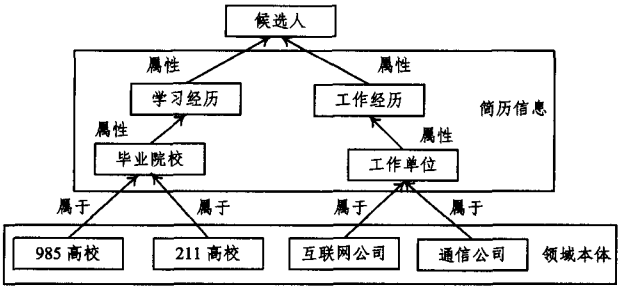

图 3　本体设计

3.1.2　简历知识库的构建

简历知识库的构建是基于关系型数据库到 RDF 数据模型的映射完成的。候选人的简历信息一般以表格的形式存储在关系型数据库中，需要将这一部分信息转换成 RDF 数据模型并导入到系统中，这是知识库的主要来源。根据系统设计的本体，本文通过两步完成关系型数据库到 RDF 数据模型的映射。

第一步：首先从原有的关系型数据库中提取有用信息，例如候选人的姓名、年龄、学习经历和工作经历等；再将这些信息转换成 XML 文件；然后对 XML 文件中的一些非结构化的信息（例如“工作单位和职位”）进行分词。

第二步：根据上面设计的本体中的类、实体和属性，通过递归算法将 XML 文件转换成 RDF 文件。如果 XML 文件中的节点有子节点，该节点就会生成对象属性和实体，然后依次递归地遍历子节点；否则，只创建一个数据属性。

3.1.3　领域知识库的维护

领域知识库是动态更新的，系统允许编辑类、实体和属性。领域知识库的维护包括领域知识的批量导入以及基于实体匹配算法的领域知识的提取，具体流程如图 4 所示。当系统批量导入领域知识时，新类会被创建，同时代表该领域知识的实体也会被导入到类中。新导入的实体和知识库原有实体之间的歧义性需要消除。如果新类中的实体和其所有父类的某个实体匹配成功，这个已经存在的实体就会直接添加到这个新类中；如果没有匹配成功，就需要为这个新类创建一个这样的实体。领域知识的提取是基于实体的匹配算法，这是对简历进行语义分析和生成人员标签的基础。一些系统需要的领域知识可能外部不存在或者不完整，就需要从简历知识库中提取。可以从某个候选类中根据实体匹配算法提取到相应的实体，从而扩充原有类。这其中也包含数据重复、不一致的问题。

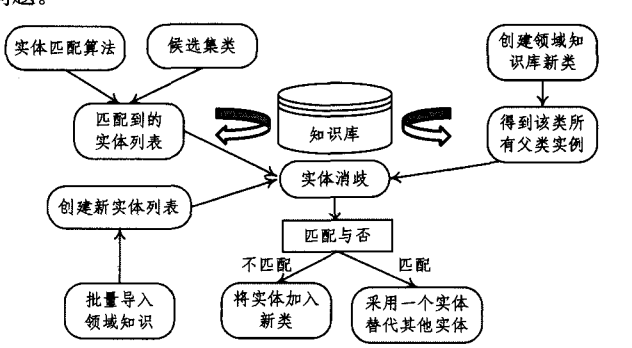

图 4　领域知识库维护

3.2　标签的自动生成

通过以上领域知识辅助信息提取的方法，扩充领域类，再

对系统中的人进行语义分析，自动生成语义标签，流程如图 5 所示。例如，需要生成“具有互联网公司研发工作经历的人”的人员标签，系统除了从外部批量导入小部分互联网公司名单和研发职位外，还需要根据实体匹配算法从简历信息中提取人员工作单位和职务信息，扩充互联网公司领域知识，构建完整的互联网公司类和研发职务类。然后借助于领域知识，对人员以前的工作单位及工作职位进行语义层面的分析，工作单位实体属于互联网公司类，工作职务实体属于研发职务类，这才算具有互联网公司研发工作经历。采用 SPARQL 查询到这些人员，生成相应标签。

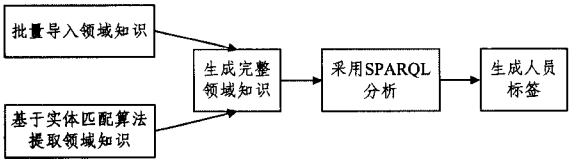

图 5　标签的自动生成

4　性能优化

性能是大规模的简历分析搜索系统面临的一个重要问题。大量的简历数据以 RDF 形式组织，而 Jena 数据库底层以三元组的形式存储到 MySQL。1 万人的简历数据存储到 MySQL 中，其三元组数量能够达到 40 万行。搜索过程中，SPARQL 语句首先会被 SPARQL Search Engine 翻译成 SQL[3]，并产生不少的 join 操作。在大规模的三元组中，过多的 join 操作会严重影响系统的查询性能。为此，本文通过改写 SPARQL 语句以及改变原有的底层存储结构来减少 join 操作，有效提高了性能。同时，本文还设计了一套多层次缓存，大大减少了查询时间。

4.1　SPARQL 查询优化

查询语句的优化是性能优化的第一阶段，其目的是重新排列元组的顺序：将最具有选择性的三元组放在最前面，将 FILTER 放在最前面，最小化 join 操作的数目。依照这些原则，本文重写了 SPARQL 查询语句，但是查询时间还是不能满足目标。

4.2　存储优化

本文采取了属性表和垂直存储的混合存储方式。属性表是一个 n 列的用于存储 RDF 数据的表，每一列存的是属性值，具有相同主语的属性值被存放在了一行。这个划分常常用到聚类算法，保证了表中不会存在过多的空值。简历系统的主要数据是基于关系型数据库的，被划分成属性表时，不会很稀疏。然而，直接将属性表运用到系统中会存在多值的情况。例如，一个候选人可能属于多个类，他有 985 高校毕业的、211 高校毕业的、具有互联网公司工作经历等多个标签。在这种情况下，候选人的标签列有多个值，依次将它们列入表中，这就是采用的垂直划分的方式。

```
SELECT DISTINCT ?x ?infoWHERE
{
  ?x p. 1:Foundation_Property ?info.
  ?x rdf:type
  p. 1:985UniversityCandidate_Class.
}
```

图 6　SPARQL 语句优化后

4.3　多层次缓存机制

本文还提出了一个多层次缓存优化机制。第一层预处理候选人的基本信息并且把它提前存为候选人的对象属性。第二层对候选人提前进行分类，打上不同的标签。如图 6 所示，其简化了 SPARQL 语句，减少了翻译成 SQL 的 join 操作。第三层是当启动服务时，将查询结果加载到内存中，并建立好索引。一旦用户发送了一个查询请求，服务器会先在内存中搜索。如果命中就直接返回结果，否则，应用服务器会将请求发送到数据库服务器进行查询，再将结果依次返回给用户。这个优化的多层缓存机制通过预处理及提前查询，提供了更好的查询性能，也避免了向数据库服务器发送过多的请求。

5　实验部分

本文的实验环境中，候选人总数达到 12000，三元组数目达到 40 万行。实验记录了用户在搜索框中输入类似“谁是 985 高校毕业的”的查询条件的时刻和返回查询结果的时刻，以计算查询时间。通过大量的实验发现，三元组的数量以及查询结果的数量等因素都会影响到系统的性能。

表 1 和图 7 示出了两个影响查询性能的主要因素，一个是查询结果数量，另一个是搜索三元组的数量空间。图 8 表明，随着查询结果增加，三元组数量增加，查询时间也会显著增长。3 次性能优化之后，查询语句减少了，查询时间也缩减到了 1s 左右，优化前后的性能参数如表 2、表 3 和图 8 所示。其中 Q1 为搜索 985 高校毕业，Q2 为大学专业并为理科的，Q3 为互联网公司研发工作经历，Q4 为通信公司管理工作经历；Opt1 为 SPARQL 查询语句优化，Opt2 为 SPARQL 查询语句和 RDF 存储优化，Opt3 为 SPARQL 查询语句、RDF 存储和缓存优化。另外，系统还实现了内存的混合查询，这些查询的性能在 3 次优化之后有了明显提升。

表 1　影响性能因素

	查询结果数量		搜索三元组数目范围
Q1	249	优化前	400000
Q2	3109	Opt1	400000
Q3	1414	Opt2	12000
Q4	3375	Opt3	4000 以下

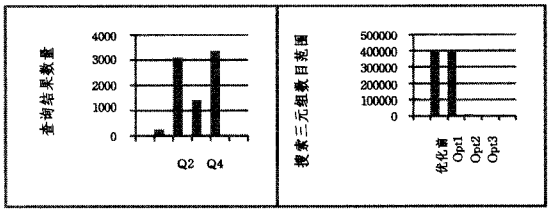

图 7　影响性能的主要因素

表 2　优化前后语句比较

	Q1 语句个数	Q2 语句个数	Q3 语句个数	Q4 语句个数
优化前	11	12	16	18
Opt1	11	12	16	18
Opt2	2	2	2	2
Opt3	2	2	2	2

表 3　优化前后查询时间比较

	Q1 查询时间(s)	Q2 查询时间(s)	Q3 查询时间(s)	Q4 查询时间(s)
优化前	3.1	51	18	49
Opt1	3	50	17.5	46
Opt2	1	8	5	7
Opt3	0.5	0.7	0.6	0.9

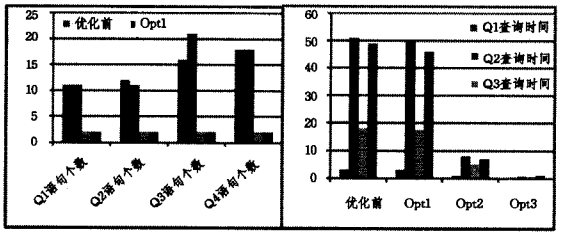

图 8 优化前后性能比较

6 相关工作

RDF[4]数据模型是一种知识表示模型，由形如$\langle s,p,o\rangle$的三元组语句集合构成。目前，世界各国各组织建立的知识库多达 50 余种。其中，代表性的知识库有 KnowItAll、TextRunner、NELL、Probase、Satori、PROSPERA、SOFIE 以及一些基于维基百科等在线百科知识构建的知识库，如 DBpedia、YAGO、Omega 和 WikiTaxonomy。除此之外，一些著名的商业网站、公司和政府也发布了类似的知识搜索平台，如 Evi 公司的 TrueKnowledge 知识搜索平台，Google 的知识图谱(knowledge graph)、Facebook 推出的类似的实体搜索服务 Graph Search 等。在国内，中文知识图谱的构建也有大量的研究和开发工作，代表性工作有上海交通大学最早构建的中文知识图谱平台 zhishi. me、百度推出的中文知识图谱搜索、搜狗推出的知立方平台。

知识库的构建主要包括知识库构建、多源知识融合与知识库更新。知识库构建是构建几个基本的构成要素，包括概念、实例、属性和关系。现有的知识库构建方法主要分为 3 类：直接编辑知识、利用大众智慧、自动或半自动知识抽取。直接编辑知识是指从事知识库构建的研发人员或者数据标定人员把自己的知识写成计算机能够处理的格式，但受时间和经济成本的约束，很难实现大规模知识库构建。利用大众智慧是指利用互联网众包机制，设计合理的激励机制，使互联网用户利用业余时间向知识库贡献一些信息。自动或半自动知识抽取方法是利用自动或半自动的算法，从非结构化和半结构化文本中提取知识。多源知识的融合是为了解决知识的共享与复用问题，避免从头开始构建一个知识库。其核心问题是解决前后导入知识的冲突、冗余以及不一致等问题。

大部分的 RDF 存储系统采用关系型 DBMS 直接以三元组存储，例如 Jena[5]、Sesame[6]、3store[7]和 Rstar 等。这种方式虽简单直接，但会产生一个相当大的表。另外，URL 和文本一般都比较长，很多 RDF 存储系统，例如 Oracle 和 Sesame 只会存储缩略的字符串。基于关系数据库的存储方式的主要优点在于它具有成熟稳定的关系型查询引擎，这些功能能被 RDF 复用。另外，由 SPARQL 翻译成的 SQL 能被关系型数据库直接优化，并返回查询的结果。然而，真实应用往往存在很多 join 操作。由于存在索引，如果主语、谓语和宾语是常量，这样的查询语句更具选择性，join 操作的代价就会小很多。而实际情况是，它们不太具有选择性，再加上三元组数量巨大，这就导致查询耗时。于是，Jena2[8]和 Oracle 改变了存储模式，采用属性表[9]，减少了大量的 join 操作。但是这个方式需要 RDF 数据是结构化的，否则会产生很多稀疏表。Wilkinson 指出，为了避免表的稀疏性，大量的属性表可能会被创建，相应的跨表操作会大大降低查询速度。同时，属性表还存在大量多值问题。AbadiDJ 指出，SW-Store[10]采用垂直分割的方式，避免了属性表的稀疏以及多值的问题，但是却增加了 join 操作。

结束语 语义搜索技术通常被用于智能数据的管理与查询。本文设计的简历搜索系统是一个有关人力资源的典型语义网的应用。它主要的功能是通过领域知识库辅助信息提取的方法，对简历信息进行语义分析和标签自动的生成。它为用户提供了简单的语义搜索接口，从而能够精准高效地从大量的简历中筛选到合适的人选。另外，在大规模人员的情况下，本文提出了多层次缓存方法，其能有效实现性能的提高。

我们未来的工作还会集中在性能优化上。随着候选人的数量的急剧增加，系统的查询性能会受到极大的影响。分布式系统(例如流行的 NoSQL 存储)或并行计算都将用于处理不断增长的数据。

参 考 文 献

[1] Tim B-L, Hendler J, Lassila O. The Semantic Web[OL]. http://www. scientificamericaon. com/article/the-semantic-Web/

[2] Harris S, Shadbolt N. SPARQL query processing with conventional relationaldatabase systems[C]// Proceedings of the 1st International Workshop on ScalableSemantic Web Knowledge Base Systems (SSWS 2005). 2005:235-244

[3] Elliott B, Cheng E, ThomasOgbuji S, et al. A complete translation from SPARQLinto efficient SQL[C]// Proceedings of the 2009 International Database Engineering& Applications Symposium (IDEAS 2009). 2009:31-42

[4] Klyne G, Carroll J J, McBride B. Resource description framework (RDF):Conceptsand abstract syntax[R]. W3C recommendation, 2004

[5] Carroll J J, Dickinson I, Dollin C, et al. Jena: implementing the semantic web recommendations[C]// Proceedings of the 13th International World Wide Web conference on Alternate Track Papers. ACM, 2004:75-83

[6] Broekstra J, Kampman A, van Harmelen F. Sesame: a generic architecture for storingand querying RDF and RDF Schema[C]// Proc. of the International Semantic Web Conference (ISWC). 2002:54-68

[7] Harris S, Gibbins N. 3store: efficient bulk RDF storage[C]// Proc. of the InternationalWorkshop on Practical and Scalable Semantic Systems (PSSS). 2003

[8] Wilkinson K, Sayers C, Kuno H, et al. Efficient RDF storage and retrievalin Jena2[C]// Proc. of the International Workshop on Semantic Web and Databases(SWDB). 2003:131-150

[9] Wilkinson K. Jena property table implementation[C]// Proc of the 2nd Int Workshopon Scalable Semantic Web Knowledge Base Systens. 2006:35-46

[10] Abadi D J, Marcus A, et al. SW-Store: a vertically partitioned DBMS for SemanticWeb data management[J]. VLDB Journal, 2009, 18(2):385-406