

基于规则引擎的个性化主题网页爬虫的研究

赵思佳, 尹 婷

(中南大学 信息科学与工程学院, 湖南 长沙 410083)

摘 要: 目前互联网上的信息正在飞速的增长, 人们主要依靠搜索引擎查找信息, 随着专业化不断加强, 垂直搜索引擎成为人们新的工具, 但构建专业化搜索引擎过程比较复杂。为了解决垂直搜索引擎中主题爬虫配置不灵活的问题, 采用在爬虫上集成规则引擎的方法, 通过规则库来控制爬虫运行, 并且使用可扩展度较高的开源爬虫项目 Heritrix 和开源规则引擎项目 Drools, 构建配置方便、灵活度高的个性化爬虫, 从而将原先主题爬虫的设置从紧耦合转变成了松耦合, 降低了用户配置难度。

关键词: 规则引擎; 主题爬虫; 搜索引擎

中图分类号: TP31

文献标识码: A

文章编号: 1673-629X(2011)03-0056-04

Research of Personalization Theme Crawler Based on Rule Engine

ZHAO Si-jia, YIN Ting

(Institute of Information Science and Engineering, Central South University, Changsha 410083, China)

Abstract: Information on the internet is now rapid growth, people mainly rely on search engines to find information, continue to strengthen as the specialized, vertical search engines become the new tool, but the process of building specialized search engines is more complex. In order to solve focused crawler is not configured flexible on vertical search engines, adopt an integrated rules engine in the reptile on the way to control the reptiles through the rule base running, and use a higher degree of open source scalable Heritrix crawler project and open source rules engine project Drools, easy to build configuration, and high flexibility of individual reptiles, which will set the original theme from the tight coupling reptiles turned into loosely coupled, reducing the user configuration difficult.

Key words: rules engine; subject crawler; search engine

0 引 言

如今是一个信息膨胀的社会, 用户普遍习惯通过互联网查找信息, 2008 年 Google 公布其最新索引网页数量: 1 000 000 000 000 个网页, 一兆(百万的平方)的网页数, 数量比银河系的星体还多出一倍, 并且每天以 1 百万的数量进行增加^[1]。

正是有 Google 这样的搜索引擎的出现, 给人们提供了一个可以检索互联网资源的方式, 但是目前类似于 Google 这样的通用搜索引擎在搜索结果中会出现大量的搜索噪音, 因为它是将互联网的信息尽可能的收集, 主要考虑的是查全率, 而且给出的搜索结果只能是网页地址, 而不是具体的信息, 这样的搜索引擎并不适合企业搜索。对于企业搜索而言, 它需要的是同行业里相关的数据, 并且是格式化的, 如查看某个商品在互联网上的报价, 互联网出售票务的信息, 所以相对来

说企业搜索具有很强的针对性。

这个时候垂直搜索引擎就出现了, 垂直搜索引擎主要是利用主题爬虫抓取符合某个主题的网页, 再从非结构化的网页中抽取出结构化的数据进行存储, 使得用户最终进行搜索时得到的不是网址, 而是最终的信息。现在构建自己的垂直搜索引擎需要了解所要抓取网页的格式, 再针对网页格式编程抽取其中的信息。但是, 主要是要抓取其他不同网页或原抓取网页格式变化了, 就需要重新对搜索引擎进行重新编程, 才能满足用户需要。目前一项新的技术正在快速发展中, 就是规则引擎, 基于这个技术的特点, 提出将规则引擎用于网页爬虫上, 可以得到一种用户能灵活配置、个性化的搜索引擎^[2]。

文中主要研究了搜索引擎中一种基于规则引擎的动态主题爬虫, 它能根据用户设置的规则实时调整爬虫的爬行方式、爬行内容等方面, 从而提高了爬虫对主题网页抓取的准确率和可配置性。

收稿日期: 2010-07-24; 修回日期: 2010-10-27

基金项目: 信息产业部电子发展基金项目(信部运[2006]634号)

作者简介: 赵思佳(1983-), 男, 湖南衡阳人, 湖南环境生物职业技术学院讲师, 研究方向为计算机网络、信息系统。

1 爬 虫

搜索引擎是互联网上查找信息的必备工具, 它一

般是由三个模块构成:爬虫模块、索引模块和检索模块。其中爬虫模块是搜索引擎中重要的一部分,它负责从互联网上抓取信息,并且定时进行更新,而索引模块是对抓取来的网页按照某种标准进行排序,提高检索速度,检索模块就是一个和用户交互的界面,用户输入要查询的内容,然后将结果进行展示^[3]。

爬虫模块一般要定时访问互联网资源,更新数据库里的资源信息,及时反映网页的更新情况,去除无法访问的链接。爬虫其实就是一个自动抓取网页程序,它通过种子网址访问互联网上的网页,并对网页进行保存,再遍历网页上的链接,将这些链接加入到下一个访问对象的队列中,以此实现自动访问网页,并下载网页内容^[4]。

1.1 爬行算法

爬虫的爬行访问方式是通过爬行算法来控制的,目前将爬虫的爬行算法主要分成深度优先、广度优先和最佳优先三种。广泛采用的是广度优先和最佳优先两种,因为深度优先容易导致爬虫陷入某个网站而出不来^[5]。

广度优先算法主要是让爬虫在抓取的过程当中,先完成当前层次的爬行,再进行下一个层次的搜索。这个算法设计比较简单,容易实现,主要用于使爬虫的爬行覆盖面广,但是这种算法在对于具有某一主题的爬行而言,就会抓取大量和主题无关的网页,使得算法效率低下^[6]。

最佳优先算法是对当前网页进行分析,预测网页中的链接与爬行主题的相关程度,只选取其中评价比较好的链接进行爬行,这样就可以在抓取网页前就对大量无关网页进行了过滤,经研究表明,这样的处理可以将无关网页数量降低 30% ~ 90%^[7]。

1.2 网页识别

前面提到的最佳优先爬行算法,其中有一个重要的环节就是网页的识别,网页的识别可以大幅度提高爬虫的抓取准确率,当然在抓取速度上会有所影响。

网页的识别算法主要归纳为三种:一是基于网络拓扑的分析算法,这种算法主要是依靠同类网页之间会相互添加链接指向对方,通过已知网页,对和其有直接或间接链接关系的网页进行评价,以此作为是否抓取的依据。其中又根据分析对象的粒度不同分为网站粒度分析算法、网页粒度分析算法和网页块粒度分析算法。二是基于网页内容分析算法,是根据网页内容与所要抓取的主题进行相关度评价,将评价高的网页进行抓取保存。这种算法也经历了发展更新,从原来只针对文本和超链接为主的网页,发展到针对结构化的数据源动态生成的页面,再到针对介于前两者之间的具有良好结构、遵循一定模式或风格可以直接访问

的网页。使得爬虫抓取的网页从静态逐步转到了动态页面^[8]。三是基于用户访问行为的分析算法,这是依据用户访问网页的行为方式来作为网页的评价标准,抓取评价较高的网页。

1.3 构建爬虫的新思路

目前常见的爬虫在爬行上主要考虑的是抓全率,其中以 Google 为例,这样就要求爬虫具有较高的爬行速度和更新速度,但是爬虫就不会考虑爬行的内容,甚至是爬行内容里包含的信息。这种搜索引擎已经被广泛使用,但是现在越来越多的用户或企业希望能从网页的内容中得到他们想要的信息,这就需要能准确分析网页。以 Heritrix 为例,Heritrix 是一个由 Java 开发的、开源的、可扩展的 Web 网络爬虫,功能十分强大,最出色之处就在于它的可扩展性,开发者可以扩展它的各个组件,来实现自己的抓取逻辑。但是如果使用 Heritrix 作为专用的爬虫,则需要对其中的某些类进行重新编写,这样无形中增加了用户构建自己的专用搜索引擎的难度,无法使其大众化。

如果将各个组件中的抓取逻辑用规则引擎来替代,可以形成一种新型的搜索引擎,这种搜索引擎将大大降低用户的定制个性化搜索引擎的门槛,提高了用户参与度,使得搜索引擎中存储的内容更加符合用户要求,更加精准,并能将其轻松推广使用。

2 规则引擎

规则引擎是人工智能领域里研究的产物,现在将它用于应用系统开发。以往在开发应用系统的时候,系统的业务规则是在需求分析阶段确定,并在开发过程中嵌入到程序代码当中。然而应用系统在使用的过程当中,业务规则可能会经常发生变化,这时每一次变更都需要重新编写源代码,如果采用规则引擎开发应用系统就可以大大提高系统的灵活度,延长系统的使用寿命^[9]。

可以把规则引擎看成是一个 IF/THEN 的高级解释器,它的任务就是将提交的数据对象与规则库里的规则进行对比,激活相符合的业务规则,再根据规则中声明的执行逻辑进行相应操作。在众多规则引擎中,Drools 是出类拔萃的^[10]。

2.1 Drools

Drools 是一个易于访问、易于调整以及易于管理企业策略的开源业务规则引擎,符合 JSR94 标准,执行速度快、效率高。Drools 规则引擎主要包括三部分:规则库(Rules)、匹配引擎(Inference Engine)和事实工作空间(Working Memory),如图 1 所示。

其中规则存在规则库(Production Memory)中,推理机要匹配的事实(facts)存在 Working Memory(工作

内存)中。事实被插入到工作内存中后,可能被修改或删除。一个有大量规则和事实的系统可能会有很多规则被满足,这些规则被称为具有冲突性。Agenda 通过(冲突决策策略)管理这些冲突规则的执行顺序。

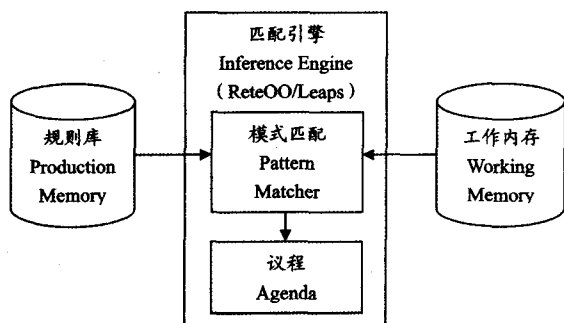


图 1 Drools 规则引擎软件结构

Drools 采用了 Rete 和 Leaps 算法来实现规则匹配,称之为 ReteOO 和 Leaps,经大量实验证明,这两种算法在进行规则匹配时具有相当高的效率^[11]。

2.2 Drools 的优点

Drools 是一个使用基于规则的方法实现的专家系统的规则引擎,更准确的说属于产生式规则(使用一种抽象结构来准确描述形式语言)系统。其主要优点是:

1) 声明式编程。

规则引擎允许你描述做什么而不是如何去做。这里的主要优点是使用规则更加容易对复杂的问题进行表述,并得到验证,使得规则比编码更容易阅读。规则系统能够解决非常困难的问题,并提供了方案怎样达到和在解决问题的方向上所作的每一个决定的原因,这个对于类似神经网络这样的智能系统来说是不容易达到的。

2) 逻辑与数据分离。

数据保存在系统对象中,逻辑保存在规则中。这根本性地打破了面向对象系统中将数据和逻辑耦合起来的局面。这样做的结果是,将来逻辑发生改变时更容易被维护,因为逻辑保存在规则中。这点在逻辑是跨领域或多领域中使用时尤其有用。通过将逻辑集中在一个或数个清晰的规则文件中,取代了之前分散在代码中的局面。

3) 速度及可测量性。

很多规则引擎采用了类似 Rete、Leaps 等算法来进行规则匹配,这些算法都具有相当高的效率,尤其是当数据改变较少时,因为规则引擎能够记得之前的匹配。

4) 知识集中化。

通过使用规则,将建立一个可执行的规则库。这意味着规则库代表着现实中的商业政策唯一对应,理想情况下可读的规则还可以被当作文档使用。

5) 工具集成。

例如可以集成到像 Eclipse 这样的开发工具上,为规则的修改与管理、即时获得反馈、内容验证与修补提供了平台,今后还将在基于 Web 的界面上进行实现。

6) 解释机制。

规则引擎还具备了审查与调试工具,通过将规则引擎的决断与决断的原因一起记录下来,规则系统提供了很好的“解释机制”。

7) 易懂的规则。

通过建立对象模型以及 DSL(域定义语言),你可以用接近自然语言的方式来编写规则。这让非技术人员与领域专家可以用他们自己的逻辑来理解规则。

3 个性化爬虫

现在知道 Heritrix 和 Drools 是在各自领域里较为优秀的开源项目,Drools 具有规则制定简单、执行效率高特点,Heritrix 具有强大的可扩展性,所以在构建个性化爬虫的时候是以 Heritrix 为基础的。

3.1 爬虫结构

在 Heritrix 工作流程中,CrawlController 是核心组件,它决定了整个抓取任务的开始与结束。当用户在控制台设置抓取任务后,Heritrix 首先构造一个 CrawlController 实例并初始化,再启动线程池和 Frontier,以便向线程池中的工作线程提供抓取用的 URL 链接,最开始的是设置好的入口种子链接,之后是分析出来的新链接。紧接着抓取线程工作开始,不断循环整个处理器链。除非用户暂停或终止,否则直到无链接可抓,控制器才认为任务执行完成。

在整个抓取流程中,分别有 BdbFrontier、PreProcessor、Fetcher、Extractor、Writer、PostProcessor 六个步骤,其中 BdbFrontier 是用来处理 URL,负责分配下一个被处理的 URL;PreProcessor 是用来处理抓取前的准备工作;Fetcher 是用来处理网络传输协议;Extractor 是用来处理服务器返回的内容;Writer 则是将抓取的信息存储到磁盘上;PostProcessor 是用来从网页中提取 URL,并将符合条件的添加到抓取队列中^[12]。

Heritrix 抓取流程大致如图 2 所示。

整个处理流程就像一个流水线,流水线上每一个环节之间又相互关联、依靠,但每个环节又相对独立,都有自己的任务,都有完成自己任务的方式,并且可以进行扩展来完成指定功能,但是要完成指定功能就要对这个步骤对应的代码进行重新编程,如果能将每个环节上都整合规则引擎,通过编写规则库文件来指定完成功能,将大大降低扩展每个组件的难度。

3.2 整合规则引擎

可以为 Heritrix 工作流的各环节制定相应规则库,

并且在环节中加入规则引擎进行判断,从而达到灵活设定的目的。在各个环节要整合规则引擎,其实就是将规则引擎在其中进行实现,规则引擎的实现算法如下^[13]:

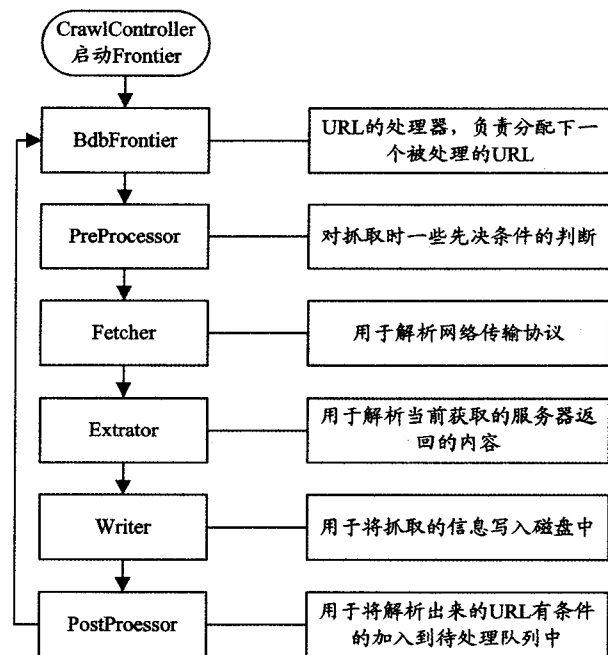


图2 Heritrix 工作流程

```
//读取 .drl 类型规则文件
businessRules = RuleBaseLoader.getInstance().loadFromReader(
    new InputStreamReader(
        com.flm.rules.example.BusinessLogicLayer.class
        .getResourceAsStream("/rulebasefile.drl")));
//启动工作内存安全会话
StatefulSession workingMemory = businessRules.newStatefulSession();
//设定工作内存添加事实的方式为事件监听
workingMemory.addEventListener(new DebugWorkingMemoryEventListener());
//添加事实
workingMemory.insert(myfacts);
//激活工作内存中的所有规则
workingMemory.fireAllRules(filter);
```

这样每个环节都采用这种方式来执行规则引擎,下面用户所要做的就是使用简单的声明式语言来制定爬虫规则,并将其保存在规则文件(.drl)中,而不需要再去重新编程而修改程序源代码,而且每次修改好规则文件后,在控制台重新激活规则库就可以使其生效,操作非常方便。

3.3 爬行算法规则

爬行算法的控制实际上就是对网页上解析出来的URL有条件加入到待处理队列中,按照何种规则添加就直接影响下一个爬行对象。通过前面 Heritrix 工作

流的介绍,其中 BdbFrontier 和 PostProessor 就是用来设定爬行规则的地方。

BdbFrontier 主要是用来分配下一个被处理的URL,PostProessor 则是判断解析得到的URL是否添加到处理队列中。

根据规则引擎的规则编写,如果要去掉解析中得到的部分扩展名的链接,可以写出这样的规则文件:

```
#去除所有扩展名为 .zip、.exe、.rar、.pdf、.doc 的链接
rule "Removal .zip URL"
when
    #获得 URL 链接扩展名
    String url = caUri.toString()
    thisexten = url.endsWith(
    #判断是否为符合的扩展名
    $thisexten: url.endsWith(extension == ".zip")
then
    #处理 URL
    getController().getFrontier().noschedule($thisexten);
end
```

上面列出的这个规则文件部分内容,实际就是判断URL链接是否指向zip文件,如果是,就对该URL执行不添加到队列处理。个性化爬虫的使用者可以通过简单的文本形式自行定义爬虫规则,使爬虫真正按用户的意思进行工作,当然以后的发展是要将规则文件的可视化编写加入进来,让使用者只需要点击鼠标就能生成规则库,从而进一步简化用户的操作。

4 结束语

文中通过对常见的通用搜索引擎中的爬虫工作原理进行了分析,得出目前搜索引擎存在重爬全,而轻爬准的问题,特别是针对企业搜索的应用上,不能进行灵活配置管理等问题,之后介绍了目前研究热门的规则引擎技术,并提出将规则引擎应用于搜索引擎的新思路,从而得到一种用户可自行灵活定义的爬虫,用以抓取符合用户定义规则的网页,使得构建新的垂直搜索引擎不再需要重新编写程序,进一步提高了搜索引擎的个性化、大众化,为搜索引擎的云计算应用提供了新的途径。

目前只是初步尝试将规则引擎集成到搜索引擎中的爬虫,其实可以将规则引擎更加深入整合到搜索引擎的各个环节,加上用户控制和可视化规则编辑方式,就可以成为一种新的人人参与的网络应用,从而达到云计算的应用高度。

参考文献:

- [1] 陆宜梅. Web 搜索技术现状分析[J]. 沈阳大学学报, (下转第63页)

足轻重地位,最后着眼于 MAC 层的邻居发现方法。邻居发现方法的不同,影响网络扫描次数,进而影响网络能耗。文中提出了一种邻居搜索方法(NSM),首先,借助于信标帧发布邻居信息,在信标帧中扩展邻居域,在邻居域携带 SDU,为保证邻居搜索的高效,邻居域仅包含直接父节点的 SDU,而且 SDU 中采用偏移时间记录同步信息。在此基础上,设计了邻居搜索方法的详细流程,坚持冗余连接保持和能效优先的基本原则,降低了网络扫描次数。基于 NS-2 仿真平台,对纯 S-MAC 协议和使用 NSM 方法的 S-MAC 协议在不同的信标发送频率下进行了仿真实验,实验结果表明:当信标发送频率较低时,使用了 NSM 方法的 S-MAC 协议的网络功率低于纯 S-MAC 协议的网络功率,当信标发送频率较高时,两者的网络功率趋于一致。可见,文中提出的邻居搜索方法能够降低低占空比网络的能耗,在低占空比无线传感网应用领域有实用价值。

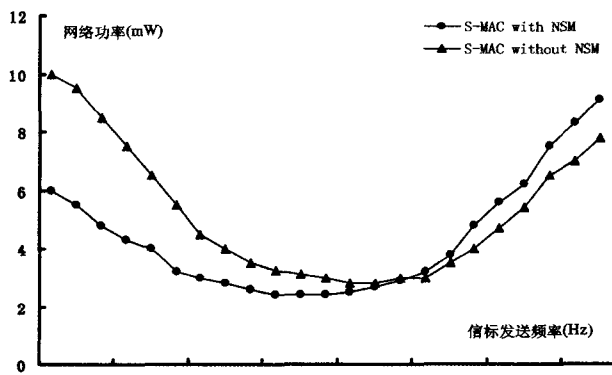


图4 网络功率-信标发送频率关系曲线

参考文献:

- [1] Callaway E H. Wireless Sensor Networks: Architectures and Protocols[M]. San Diego: CRC Press, 2003.
- [2] Roundy S, Wright P K, Rabaey J. A study of low-level vibrations as a power source for wireless sensor networks[J]. Comput. Commun., 2003, 26(11): 1131-1144.
- [3] 蹇强,龚正虎,朱培栋,等.无线传感器网络 MAC 协议研究进展[J].软件学报,2008,19(2):389-403.
- [4] Ruzzelli A G. Protocol assessment issues in low duty cycle sensor networks: The switching energy[C]//In: Proc. of the IEEE Int'l Conf. on Sensor Networks. [s. l.]: IEEE Computer Society, 2006:136-143.
- [5] 苏俊,胡访宇.无线传感器网络 SMAC 协议的节能改进[J].计算机工程,2009,35(5):106-107.
- [6] Chatterjea S, van Hoesel L F W, Havinga P. An application-tailored MAC protocol for wireless sensor networks[C]//In: Proc. of the Int'l Workshop on Wireless Ad-hoc Networks (IWWAN 2005). London:[s. n.], 2005.
- [7] 司宏林,王晓蔚.无线传感器网络多信道的 MAC 层协议[J].计算机技术与发展,2006,16(3):232-234.
- [8] 石为人,冯会伟,唐唐建.一种无线传感器网络 MAC 层协议设计与实现[J].计算机科学,2009,36(7):60-63.
- [9] Staub T, Bernoulli T, Anwender M, et al. Experimental lifetime evaluation for Mac Protocols on real sensor hardware[C]//In:Proc. of the ACM Workshop on Real-World Wireless Sensor Networks (REALWSN 2006). Uppsala: ACM press, 2006:25-29.
- [10] 刘兰军,张晓彤,王沁,等.一种基于分簇结构的无线传感器网络混合调度 MAC 协议[J].计算机科学,2008,35(9):65-71.
- [11] 徐雷鸣,庞博,赵耀.NS 与网络模拟[M].北京:人民邮电出版社,2003:32-58.
- [12] 杨风,李强,朱灵波.一种无线传感器网络跨层能量模型及能耗计算[J].计算机工程与应用,2008,44(2):172-176.
- [1] Callaway E H. Wireless Sensor Networks: Architectures and Protocols[M]. San Diego: CRC Press, 2003.
- [2] Chakrabarti S, van den Berg M, Dom B. Focused crawling: a new approach to topic-specific Web resource discovery[J]. Computer Networks, 1999, 31(11-16): 1623-1640.
- [3] 杜亚军.搜索引擎智能行为的研究及实现[D].成都:西南交通大学,2005.
- [4] 白坤,耿国华.基于 Lucene/Heritrix 的垂直搜索引擎的研究与应用[J].计算机应用与软件,2009,26(1):212-215.
- [5] De Bra P M E, Post R D J. Information retrieval in the World - Wide web: making client - based searching feasible[J]. Computer Networks and ISDN Systems, 1995, 27(2):183-192.
- [6] 贺晟,程家兴,蔡欣宝.基于模拟退火算法的主题爬虫[J].计算机技术与发展,2009,19(12):55-58.
- [7] 刘汉兴,刘财兴.主题爬虫的搜索策略研究[J].计算机工程与设计,2008,29(12):3160-3162.
- [8] 袁浩,黄烟波.网页标题分析对主题爬虫的改进[J].计算机技术与发展,2009,19(6):22-24.
- [9] 陶晓俊,朱敏.基于规则引擎的企业服务开发模式[J].计算机技术与发展,2008,18(2):115-118.
- [10] 刘伟.Java 规则引擎——Drools 的介绍及应用[J].微计算机应用,2005,26(6):717-721.
- [11] Phalp K T, Henderson P, Walters R J, et al. RolEnact: role-based enactable models of business processes[J]. Information and Software Technology, 1998, 40(3):123-133.
- [12] 希顿,李纯,童兆丰,等.网络机器人 Java 编程指南[M].北京:电子工业出版社,2002.
- [13] 王李军,陶明亮,张曙,等.面向业务规则引擎研究[J].计算机工程,2007,33(24):52-56.

(上接第 59 页)

2006, 18(2): 34-36.