垂直搜索引擎爬虫系统DIPRE算法及改进

赵 君

(武汉设计工程学院 信息工程学院,湖北 武汉 430205)

摘 要:针对垂直搜索引擎中精确抽取网页中特定字段的问题,对DIPRE算法进行了研究和改进。阐述了DIPRE算法在垂直搜索引擎中的重要作用,探讨了DIPRE算法在抽取复杂结构网页时的不足,并提出了改进,包括种子定位方式,将单模匹配扩展成多模匹配并引入定位索引,再根据已有技术对改进后的算法进行了实验验证。结果表明,改进后的算法在精度和效率上都符合预期。

关键词:垂直搜索引擎;DIPRE算法;种子定位;单模匹配;多模匹配;定位索引

DOI:10.11907/rjdk.161451

中图分类号:TP312 **文献标识码**:A **文章编号**:1672-7800(2016)008-0030-03

0 引言

垂直搜索引擎是针对某一特定领域、人群或需求提供的信息检索服务,因此垂直搜索引擎的爬虫(Spider)在抽取数据时应该具有相当的选择性。DIPRE(Dual Iterative Pattern Relation Extraction)是Google创始人之一Sergey Brin针对抽取互联网上特定格式或类型的数据而提出的一种算法,由于垂直搜索引擎具有较强的专业性和针对性,因而DIPRE算法在垂直搜索领域里具有较为广阔的应用前景,但随着Internet上的信息量呈指数级增长,网页结构越来越多样化,利用DIPRE算法抽取数据无论是在广度还是在精度上都已遇到瓶颈[1],如何在发挥DIPRE算法优势的基础上弥补其不足成为一个值得研究的问题。

1 DIPRE算法局限性

将Internet上的所有网页信息定义成一个数据库D,将要抽取的数据定义成一个关系R,R是D中的一个表[2],包含特定目标字段$f_1,f_2,\cdots,f_n$,则可以定义关系$R=(f_1,f_2,\cdots,f_n)$。垂直搜索引擎爬虫的任务就是根据关系检索D中的数据,合并成一个元组(Tuple)。爬虫在检索过程中会碰到查全率(Recall Rate)RR、查准率(Precision Rate)PR和错误率(Error Rate)ER的问题[3]。设实际检索的关系为R',则$RR=|R'\cap R|/|R|$,$PR=|R'\cap R|/|R'|$,$ER=|R'-R|/|R'|$。DIPRE算法实现步骤如下:①挑选若干个种子s_1,s_2,…s_i…,s_n,定义种子集$S=\{s_i|s_i$满足关系R,$1\leqslant i\leqslant n\}$;②利用爬虫检索D,搜索种子页面Ps;③搜索Ps中包含种子集S的代码段,该代码段应包含足够长的前后缀信息;④用通配符".*?"替换掉所有的种子,形成模式(Pattern)。定义种子s_i的前缀为Pf_i,后缀为Sf_i,则模式$Pt=Pf_1.*?Sf_1Pf_2.*?Sf_2...Pf_i.*?Sf_i...Pf_n.*?Sf_n$;⑤爬虫利用生成的模式Pt检索D,获取元组。步骤⑤中如果元组的ER值超过10%[4],则该Pt的生成是失败的,因而必须重复步骤②~⑤。

上述迭代过程中有一个重要环节就是Pt的生成,Pt是爬虫根据Ps中的关系R生成的,而Ps具有不确定性,这是因为D是一张有向图<V,E>,当爬虫从入口爬行到第一张满足R的页面时,所经历的弧不一样,Ps也会不同[5],导致Pt变化,但爬虫一次只能使用一个Pt,这就是单模匹配模式。Pt与R的复杂度有关[6],当R较复杂时,Pf_i和Sf_i中可能会出现变量(噪声)干扰Pt,使之不能与目标页面的关系匹配,降低了RR值,DIPRE算法即告失效。

2 DIPRE算法改进

一张HTML页面可以描述成一个DOM节点树[7],根节点是<html>标签,为了论述方便,将根节点命名为root,将root下的子节点<head>、<body>命名为N1、N2,N2下的子节点命名为N21、N22、N23…依此类推,可以画出种子页面的节点树状图如图1所示。

作者简介:赵君(1980—),男,湖北荆州人,硕士,武汉设计工程学院信息工程学院教师,研究方向为网络数据挖掘、大数据分析与处理。

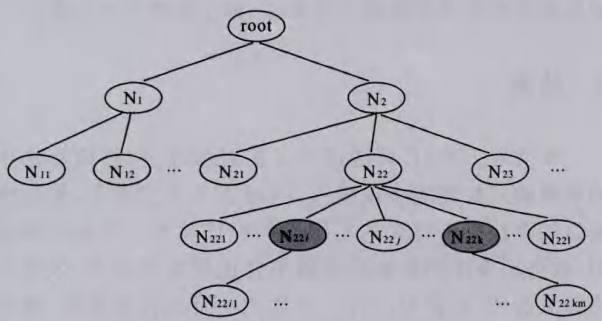

图 1　种子页面 DOM 节点树状图

图 1 中实心圆圈处的节点为种子节点,节点 N22 下的子节点中至少有两个(N22i 和 N22k)包含种子,这些节点不是连续的,它们之间还包含非种子节点如 N22j。定义节点 N 的内容为<N>,如果<N22j>不包含噪声,则根据 DIPRE 算法,将非种子节点作为上下文嵌套到 Pt 中,则 Pt=Pf_{N22i}. * ? Sf_{N22i}…<N22(i+1)>…<N22j>…Pf_{N22k}. * ? Sf_{N22k},理论上模式越长,ER 值越低,但如果没有人工干预则很难区分噪声。因此 Pt 越长,包含噪声的概率就越大,RR 值就会越低[8]。在改进的算法中,种子页面改为由人工定位,文献[9]已经实现了鼠标定位页面元素,为 DIPRE 算法的改进提供了技术基础。将单模中的 Pt 拆解成多个子模式,针对 R 中的字段单独检索,并将数据合并成元组,这就是多模匹配模式。在这种模式中,Pt_i 是 f_i 的函数,Pt=$\{Pt_i | Pt_i=f(f_i), 1\leqslant i\leqslant n\}$。多模匹配可以规避噪声的干扰,提高 RR 值。但它将一个有向序列分解成了无序集合,每一个模式的长度大大缩短,因此在提高 RR 值的同时也提高了 ER 值。为更好地说明该问题,将图 1 中节点 N22 的子节点明细标示在表 1 中,如果一个节点为种子节点,则显示前缀、后缀与种子内容,否则仅显示节点内容。

表 1　N22 子节点

节点名	前缀	后缀	是否种子	种子/节点 内容
N221	×	×	否	<N221>
N22(i−1)	×	×	否	<N22(i−1)>
N22i	Pf_{N22i}	Sf_{N22i}	是	s_i
N22(i+1)	×	×	否	<N22(i+1)>
…	…	…	…	…
N22(k−1)	$Pf_{N22(k-1)}$	$Sf_{N22(k-1)}$	是	s_{k-1}
N22k	Pf_{N22k}	Sf_{N22k}	是	s_k
N22(k+1)	$Pf_{N22(k+1)}$	$Sf_{N22(k+1)}$	是	s_{k+1}

表 1 中 N22i 的两个兄弟全是非种子节点,很显然,为了降低 ER 值,在生成模式时 N22i 的前缀应该尽量向<N22(i−1)>延伸,后缀尽量向<N22(i+1)>延伸;而 N22k 的兄弟全是种子节点,因此前缀只能尽量向 $Sf_{N22(k-1)}$ 延伸.后缀尽量向 $Pf_{N22(k+1)}$ 延伸,如图 2 如下。

在延伸时,如果<N22(i−1)>、<N22(i+1)>、$Sf_{N22(k-1)}$ 或 $Pf_{N22(k+1)}$ 存在噪声,则会遇到新的问题。本文要探讨的是不向兄弟节点延伸的情况。在图 2 中,字段 f_i 对应的 Pt_i 只与 s_i 的前后缀相关,如果存在两组或以上完全相同的前后缀,则爬虫会因为无法区分字段而导致检索失败,在此需引入一个新的因子:定位索引,该索引类似于数组中的下标。假设所有种子的前后缀都相同,则用一个模式 Pt' = Pf_{N22i}. * ? Sf_{N22i} 就可以检索出所有的数据(包括非目标数据),形成一个长度为 L 的数组 Array,则定位索引的作用就在于记录 Array 中元素与目标字段的对应关系,如图 3 所示。

节点名	前缀	后缀	种子/节点 内容
N22(i-1)	×	×	<N22(i-1)>
N22i	Pf_{N22i}	Sf_{N22i}	s_i
N22(i+1)	×	×	<N22(i+1)>
…	…	…	…
N22(k-1)	$Pf_{N22(k-1)}$	$Sf_{N22(k-1)}$	s_{k-1}
N22k	Pf_{N22k}	Sf_{N22k}	s_k
N22(k+1)	$Pf_{N22(k+1)}$	$Sf_{N22(k+1)}$	s_{k+1}

图 2　节点前后缀延伸

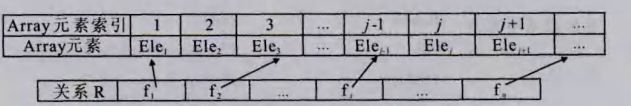

图 3　目标字段与元素对应关系

人工定位时需记录元素索引与字段的关系集合{(1,f_1),(3,f_2),…,(j−1,f_i)…},但由于 R 中的种子是有序的,因而只需记录元素索引即可,这就是定位索引,它是一个一维数组,其中的元素与 R 中的种子是一一对应的。现对传统 DIPRE 算法和改进后的算法进行效率评估。在实际检索过程中,爬虫对目标页面的爬行通常以列表(或目录)页面中锚文本的超链接为依据,所以非目标页面虽然不包含元组,但却是必不可少的[10]。设在给定的垂直搜索中,目标页面数占总页面数的比例为常数 α,爬虫对一张列表页面中有效锚文本的平均解析时间为常数 t_1;在长度为 1KB 的文本中解析长度为 10B 的模式所消耗的平均时间为常数 t_2;目标页面的文本长度为 $|H_i|$(单位为 KB),模式 P_i 的长度为|Pt|(单位为 B),为常数;爬行的页面数量为 x。根据文献[11],爬虫利用传统 DIPRE 算法爬行 x 张页面需要的时间 t_{1x} 为:

$$t_{1x} = f_1(x) = (1-\alpha)\cdot x\cdot t_1 + \alpha\cdot x\cdot\left(t_1 + \frac{|Pt|\cdot|H_i|}{1\cdot 10}\cdot t_2\right) \tag{1}$$

在同一类页面中,通过大量抽样分析,得:

$$\lim_{n\to\infty}\frac{\sum_{i=1}^{n} H_i}{n} = \text{常数} \tag{2}$$

因此,当页面数量足够多(大于 10 000)时,可以令式(1)中的 $\frac{|Pt|\cdot|H_i|}{1\cdot 10}$ =常数 β,则:

$$t_{1x} = f_1(x) = (t_1 + \alpha\cdot\beta\cdot t_2)\cdot x \tag{3}$$

在改进后的算法中,为方便测试本文仅讨论用一个 Pt' 和相应的定位索引就可以检索出一个完整元组的情况。如果爬虫检索的数组长度为 L,则爬行 x 张页面需要的时间 t_{2x} 为:

$$t_{2x} = f_2(x) = (1-\alpha)\cdot x\cdot t_1 + \alpha\cdot x\cdot(t_1 +$$

$$\frac{|Pt'| \cdot L \cdot |H_i|}{1 \cdot 10} \cdot t_2) \quad (4)$$

令式(4)中的 $\frac{|Pt'| \cdot L \cdot |H_i|}{1 * 10}$ = 常数 γ，则：

$$t_{2x} = f_2(x) = (t_1 + \alpha \cdot \gamma \cdot t_2) \cdot x \quad (5)$$

3 实验结果

实验以某大型网上书城的图书信息为检索对象，包括作者、出版社、出版时间、版次、页数共 5 个字段，此 5 个字段之间不含噪声，是测试的理想之选。使用的服务器配置如表 2 所示。

表 2 服务器配置

参数类型		配置结果
服务器系统参数	CPU	Intel Pentium G620
	RAM	2GB
	OS	Windows2003 SP2
	DBMS	MSSql Server2005
爬虫软件参数	入口地址	网上书城首页
	网页编码	GB2312
	工作线程数	10
	检索策略	广度优先
	目标页面判定策略	URL 关键词判定

以采集 40 万条数据为测试目标，采用两种算法的爬虫检索性能情况如表 3 所示。

表 3 爬虫检索性能

元组数量(单位：万条)	传统 DIPRE 算法时间(单位：min)	改进后的算法时间(单位：min)
0	0	0
5	183	205
10	362	408
15	537	600
20	713	818
25	904	1 021
30	1 087	1 217
35	1 249	1 422
40	1 459	1 612

通过抽样检测，以上检索的 ER 值均低于 10%。根据表 3 绘制出性能对比图，如图 4 所示。

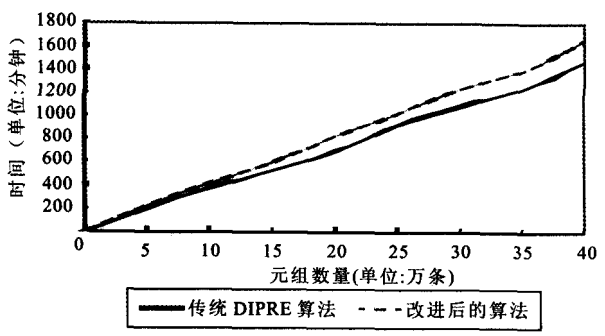

图 4 爬虫检索性能对比

由式(3)、式(5)和图 4 可知，爬虫检索页面时间和页面数量呈线性关系，其中采用传统 DIPRE 算法的斜率为 $t_1 + \alpha \cdot \beta \cdot t_2$，采用改进后算法的斜率为 $t_1 + \alpha \cdot \gamma \cdot t_2$；改进后算法的效率要略低于原算法，即 γ 值要大于 β 值。

4 结语

本文对 DIPRE 算法进行了扩展和改进，将原算法中的单模模式扩展成多模模式，同时引入定位索引，使得改进后的算法具有很强的实用性和可扩展性。实验结果表明，改进后算法的性能曲线斜率要比原算法的大，效率比原算法低，这是因为 $|\cdot|$ 过小导致无法有效过滤数据，使得 L 远大于 R 中字段数量，爬虫检索了很多无效值，降低了检索效率。在后续改进中，重点在于降低式(5)中的值，即 L 的值，这就必须使 $|\cdot|$ 达到一个合理的范围，图 2 中阐述的前后缀延伸方法是个不错的解决方案，如何控制延伸的程度则是后续研究的主要内容。

参考文献：

[1] OREN KURLAND, LILLIAN LEE. PageRank without hyperlinks [J]. ACM Transactions on Information Systems (TOIS), 2010, 28 (4): 1-38.

[2] LIU GUI-MEI. An adaptive improvement on PageRank algorithm [J]. Applied Mathematics: A Journal of Chinese Universities(Series B), 2013, 28(1): 17-26.

[3] GHOLAM R AMIN, ALI EMROUZNEJAD. Optimizing search engines results using linear programming[J]. Expert Systems With Applications, 2011, 38(9): 11534-11537.

[4] LIN LI, GUANDONG XU, YANCHUN ZHANG, et al. Random walk based rank aggregation to improving web search[J]. Knowledge-Based Systems, 2011, 24(7): 943-951.

[5] E GARCIA, F PEDROCHE, M ROMANCE. On the localization of the personalized PageRank of complex networks[J]. Linear Algebra and Its Applications, 2013, 439(3): 640-652.

[6] SHAYAN A, TABRIZI, AZADEH SHAKERY, et al. Personalized pagerank clustering: a graph clustering algorithm based on random walks[J]. Physica A: Statistical Mechanics and its Applications, 2013, 12(5): 15-24.

[7] ALEXGOH KWANG LENG, P RAVI KUMAR, ASHUTOSH-KUMAR SINGH, et al. Link-Based spam algorithms in adversarial information retrieval[J]. Cybernetics and Systems, 2012, 43(6): 459-475.

[8] LI LIAN, ZHU AI HONG, SU TAO. An improved text similarity calculation algorithm based on vsm[J]. Advanced Materials Research, 2011, 1250(225): 1105-1108.

[9] LI MIN, ZHAO JUN. Research and design of the crawler system in a vertical search engine[C]. Guilin: In Proceedings of the 2010 International Conference on Intelligent Computing and Integrated Systems, 2010: 790-792.

[10] EVANTHIA E TRIPOLITI, DIMITRIOS I FOTIADIS, GEORGE MANIS. Modifications of the construction and voting mechanisms of the random forests algorithm[J]. Data & Knowledge Engineering, 2013, 87(7): 112-118.

[11] 柳厅文，孙永，卜东波，等. 正则表达式分组的 1/(1-1/k)-近似算法[J]. 软件学报，2012，23(9)：2261-2272.

（责任编辑：孙 娟）