

基于特征串的网页文本并行去重算法

谢瑶兵

(同济大学 电子与信息工程学院, 上海 201804)

摘 要: 针对海量网页文本去重效率不高问题, 提出了一种高效的并行网页去重算法. 该算法利用 Hadoop 框架的 Map/Reduce 机制, 通过对网页文本提取特征串, 使用 Google 的 Simhash 算法对提取的特征串进行哈希映射得到相应的哈希码, 然后对产生的哈希码进行海明距离比较, 从而得到重复的网页数据. 实验表明, 与相关去重算法相比, 所提算法有效地提高了文本去重计算效率.

关键词: 搜索引擎; 特征串; 网页去重; Simhash; Map/Reduce

中图分类号: TN402

文献标识码: A

文章编号: 1000-7180(2015)02-0069-04

A Parallel Webpage Duplicate Removal Algorithm Based on Character String

XIE Yao-bing

(School of Electronic and Information Engineering, Tongji University, Shanghai 201804, China)

Abstract: Against the inefficiency of the huge amount of webpage duplicate removal method, proposes a parallel webpage duplicate removal algorithm based on character string. Using the MapReduce model in Hadoop to extract character string from webpage content, and compute the character string into hash code by Simhash algorithm. Then compare the hamming distance between all the hash code to find duplicate webpages. The Algorithm is proved to be more efficient than related algorithms based on experimental results.

Key words: search engine; character string; webpage duplicate removal; Simhash; Map/Reduce

1 引言

搜索引擎在爬取网页阶段, 会爬取到大量的冗余网页. 冗余网页的产生主要是由于互联网本身存在的一些类似转载的现象, 导致爬取的网页存在很多重复内容, 这对数据存储、索引效率以及用户体验都有很大的影响. 传统的搜索引擎对于这种现象的处理一般是采用 MD5 算法, 通过对网页正文文本使用 MD5 算法计算出文本哈希值, 然后对海量哈希文本进行比较. 然而 MD5 算法只能匹配完全一致的文本, 而冗余网页中完全一致的网页只占一小部分, 大部分是通过转载的行为产生的, 这种情况下两篇相同的文章通常只是多了一行转载信息, 却影响着 MD5 算法的结果.

针对这个问题, 文献[1]中提出了一种基于特征串的网页文本去重算法; Google 提出的 Simhash^[2]算法能对相似的文本内容产生相近的签名, 对转载的冗余文章有超强识别功能. 文献[3-6]中对近似重复网页检测和分类算法进行了深入的探讨. 本文结合特征串去重算法与 Google 的 Simhash 算法, 采用 Hadoop 框架的 Map/Reduce 的机制并行实现, 提高网页文本去重效率.

2 相关理论

2.1 基于特征串的网页去重算法

对于网页提取特征串, 首先需要对爬取下来的网页进行解析, 去除网页中的广告、导航等无用文本信息, 然后根据文章结构采用自然语句分割标志(如

收稿日期: 2014-05-20; 修回日期: 2014-06-22

基金项目: 国家自然科学基金项目(71170148); 国家科技计划课题(2012BAD35B01)

“。”、“?”和“!”等),将文章分割成一个个句子,然后提取出句子的首尾字符,最后将整篇文章提取出的字符连接成一串字符作为文章的特征串.这样得到的特征串只保留了文本内容和结构信息,简化了后续哈希映射的运算量的同时不影响去重的准确度.

2.2 Simhash 算法

传统的哈希算法只会把文本内容随机的映射为一个数字签名,相当于产生伪随机数.这种方式产生的两个签名如果相等,说明原始文本内容有一定的概率是相等的,如果不相等,只能说明原始文本内容是不相同的,并不提供两者的相似度信息,这就意味着即使原始网页只相差一行转载信息甚至一个“转”字,也会判断为不同的网页,达不到想要的去重效果.

Simhash 算法产生的哈希码,除了能判断原始文本是否相同以外,还能提供原始比较文本相差程度,以判断两者是否为转载的网页.

2.3 Map/Reduce 模型

Map/Reduce^[7]是一种处理海量数据的并行编程模型和计算框架,用于对大规模数据集的并行计算.计算运行在由多个节点组成的集群上,每个节点会被指定为 mapper 或 reducer,任务开始会把海量的输入通过 partitioner 分配给相应 mapper,计算结束后经过 combine、shuffle 和 sort,最后再进行 reduce 计算,输出结果. Map/Reduce 的伸缩性非常好,对于集群中节点的增减非常方便,而过去的大多数分布式处理框架,在伸缩性方面都与 Map/Reduce 相差甚远.通过这样一种机制,比起单机的算法实现,效率上有显著的提高.

3 并行特征串去重算法

3.1 基本思路

算法执行之始,需要对搜索引擎爬取下来的网页进行解析,获取出其正文内容,然后以自然语言分隔符“.”分割文本提取出文章的特征串,而后续的哈希映射则采用 Simhash 算法,然后对产生的哈希码比较去重.

3.2 算法并行实现

Map/Reduce 模型是以<Key, Value>的形式输入输出的,其默认输入格式为 TextInputFormat,可以将文本文件分块并逐行读入以便 Map 节点处理.所以本文算法在处理初始输入文件时,不是将一个网页写入一个文件,而是将所有的网页文本中的

换行符用空格替换,然后以“url 分隔符 content”的形式输入到同一个文件中,以实现一行文本存储网页内容的效果,方便算法实现,减少步骤.算法并行实现的具体流程如图 1 所示.

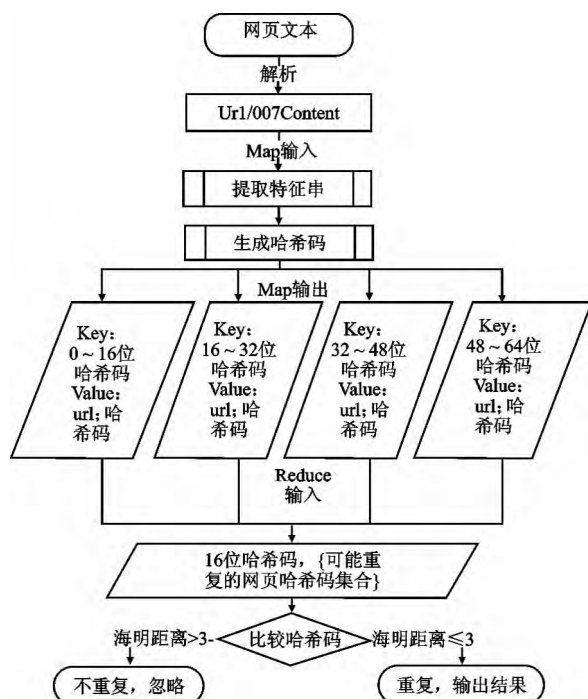


图 1 并行特征串去重算法流程

算法 Map 函数得到格式为“Url/007Content”的输入后,以分隔符“/007”分割输入,分别得到 Url 和对应的 Content,然后对 Content 以“.”分割成一个文本链表.取每个链表的首位字符作为这句话的特征,然后连接成特征串.例如本文的摘要,经过上述处理后将提取出如下特征串:“针法-该据-实效”.

得到文本的特征串后,将每个特征词映射为 64 位的哈希码,以二进制表示,则每个特征词都是由 0、1 串组成.接下来再对所有特征词的哈希码分别进行对应位加权,即哈希码的第 i 位与其他哈希码第 i 位加权,1 则加 1,0 则减 1,最后的产生结果中正的取 1,负的或 0 则取 0,由此产生输入文本整体的一个哈希映射结果,得到能反映整篇文章的一个哈希码.

得到网页正文的哈希值后,接下来需要做的就是将所有的网页哈希码进行比较.然而这种比较又不是精确匹配,只要对比的两个 64 位的哈希码中不相同的位数(海明距离)少于 3 就可以认为两个网页文本为重复的,这也正是 Simhash 算法可以查出转载的重复网页的原因.对于海量的网页数据,如果逐

一比较 64 位的哈希码,算法的时间消耗将是不可接受的.我们可以把 64 位的哈希码四等分,如果两个哈希码的海明距离小于 3,则两者等分的 4 个子哈希码中至少有一个一定是完全匹配的,如图 2 所示.

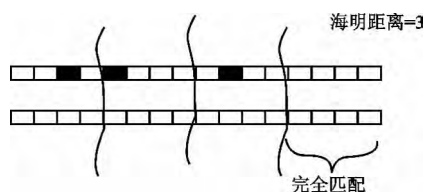


图 2 海明距离为 3 的哈希码匹配示意图

所以本算法将网页文本哈希码拷贝 4 份输出,每份的输出格式为<key:16 子哈希码,value: url; 哈希码>,至此 Map 函数结束.这样虽然存储的空间变为原理的四倍,但算法在时间消耗上得到了显著的提升.

Map/Reduce 模型在执行 Reduce 函数之前,会将 Map 输出结果根据 key 值进行排序,并将相同 key 值的输出合并提交给 Reduce 函数.本算法正是利用了这一点,将网页哈希码的 16 位子哈希码作为 key 值,这样就不需要再海量的数据中寻找相同的子哈希码,Map/Reduce 模型会直接把所有相同的子哈希码的数据交给同一个 Reducer 处理.所以 Reduce 函数的输入就是<key:16 子哈希码,value: {可能重复的网页 Url 和哈希码集合}>,接下来算法只需要将 value 的内容遍历,比较哈希码的海明距离,将海明距离小于 3 的网页的 Url 输出,就是重复的网页数据.

3.3 效率分析

由于对网页输入进行了规范,本算法整体实现只需要执行一个 Map/Reduce 任务.算法的效率取决于集群的规模,但无论如何,分布式的计算比之单机算法效率都具有不可比拟的提高.

在算法关键的比较网页文本哈希码过程中,假设网页文本数量为 n ,则对比每个网页哈希码的时间复杂度将为 $O(n)$,而网页文本数据 n 往往会达到几百万甚至上亿,在此基础上还需要对 64 位子哈希码一一匹配以找出海明距离小于 3 的网页哈希码.本算法巧妙运用 Map/Reduce 排序机制,省去了在海量数据中完全匹配子哈希码的步骤,对提高并行去重算法效率有重大影响.

4 实验

4.1 实验环境

算法由 8 台 Dell PowerEdge R720XD 计算机

组成的集群实现,其中每台计算机硬件条件如下:

PowerEdge R720 NTPM 主板;

2 颗英特尔至强处理器 E5-2650;

DIMM Blanks;

1 600 MHz RDIMMS;

256 GB RDIMM, 1 600MHz, 低电压, 双列,

x4 带宽;

1TB 7.2K RPM 近线 6 GB/ps SAS 3.5 英寸热插拔硬盘;

PERC H710p 集成 RAID 控制器, 1 GB NV 缓存;

外置 DVD-ROM 光驱.

每台计算机的软件环境:操作系统为 64 位 CentOS 6.3,Java 环境为 JDK1.7,集成开发环境为 Eclipse 3.7.2,集群搭建的是 Hadoop 1.1.2.

4.2 实验数据与结果分析

实验分为两部分,分别针对算法的准确性和算法与单机去重算法效率对比进行.

实验 1. 算法准确性

评估算法准确性有两个标准,分别为重复网页的召回率(Recall)和去重的精准率(Precision).

对于算法准确性,实验数据来源主要依靠手动获取,对两组数据进行实验,两部分数据对算法召回率和精准率的影响如表 1 所示.

表 1 去重的召回率和精准率统计

	第一组	第二组
总网页数	213	187
重复网页数	159	132
算法除去的网页数	164	130
算法正确去重数	155	130
召回率/%	97.5	98.5
精准率/%	94.5	100

由表 1 可以看出,算法的召回率和精准率都很高,说明本算法在去重方面是十分准确的.

实验 2. 算法效率对比

本实验的输入数据为 132 万因特网网页文本.

Map/Reduce 模型将计算任务划分为 Map 和 Reduce 阶段,将任务分配给 mapper 和 reducer 并行执行.任务执行的 mapper 数由集群智能分配,不能由用户自定义设置,而 reducer 数目却是可以通过 job.setNumReduceTasks() 函数设置.算法效率随着 reducer 数量的不同而不同,如图 3 所示.

由图 3 看出, reducer 数量越多算法效率越高,

然而增加到一定程度对算法效率增长并不明显,且对硬件要求较高,故针对 132 万数据源,之后的实验采用的 reducer 数量皆为 10 个。

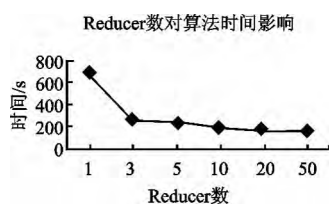


图3 Reducer 数对算法影响示意图

接下来将本算法实验结果与单机特征串结合 Simhash 算法效率进行对比,特意将输入数据从小到大复制为好几份,以对比试验结果,如图 4 所示。

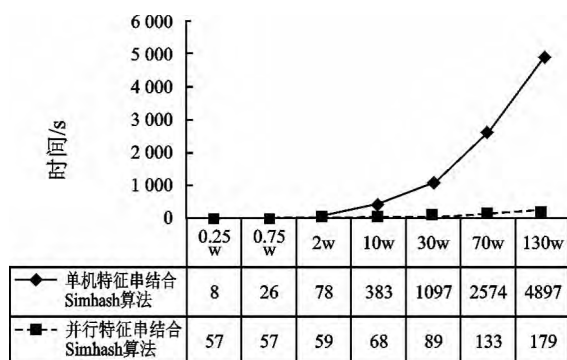


图4 并行算法与单机算法时间消耗对比图

由图 4 可以看出,在输入数据比较小的情况下(1 万到 2 万之间),单机的算法比并行算法消耗时间段,原因是并行算法无论数据量多小,Map/Reduce 模型都需要任务初始化时间。然而当输入数据量增大时,单机算法时间一直呈线性增长,而并行算法时间消耗增加并不明显^[8]。

4.3 实验结论

通过算法准确性实验以及算法效率对比实验,可以看出本算法具有切实可行性,能在海量数据集中准确找出重复文本,并且在时间消耗上,相比相关算法有显著提升,且效率提升速度将会根据数据量以及硬件的增加呈指数级增长。

5 结束语

本文提出的并行去重算法,轻松应对海量数据,

去重准确率高,可应用于搜索引擎以及其他大数据项目。

参考文献:

- [1] 吴平博,陈郡秀,马亮. 基于特征串的大规模中文网页快速去重算法研究[J]. 中文信息学报, 2003, 17(2): 28-35.
- [2] Sadowski C, Levin G. Simhash: Hash-based similarity detection[EB/OL]. [2014-04-10]. www.googlecode.com/sun/trunk/paper/Sim Hash with Bib. pdf.
- [3] Narayana V A, Govardhan A, Premchand P. Near-duplicate web page detection: A comparative study of two contrary approaches[C]// Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on. Korea; Jeju Island, IEEE, 2011: 769-776.
- [4] Zhang Y H, Zhang F. Research on new algorithm of topic-oriented crawler and duplicated web pages detection[M]. Berlin, Heidelberg, Springer, 2012: 35-42.
- [5] Gao K. The strategy on replicate and similar web collections' detecting and clustering[J]. Computer Applications in Engineering Education, 2012, 20(2): 221-231.
- [6] Kumar J P, Govindarajulu P. Near-duplicate web page detection: An efficient approach using clustering, sentence feature and fingerprinting[J]. International Journal of Computational Intelligence Systems, 2013, 6(1): 1-13.
- [7] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113.
- [8] Manyika J, Chui M, Brown B, et al. Big data: The next frontier for innovation, competition, and productivity[R]. New York; Mckinsey Global Institute, 2011.

作者简介:

谢瑶兵 男, (1989-), 硕士研究生. 研究方向为数据挖掘、云计算、电子商务. E-mail: 290765862@qq.com