第 34 卷 第 11 期 计 算 机 工 程 2008 年 6 月
Vol.34 No.11 Computer Engineering June 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)11—0270—03 文献标识码：A 中图分类号：TP391

基于 SHA-1 的邮件去重算法

张 曼，李弼程，林 琛
(解放军信息工程大学信息工程学院，郑州 450002)

摘 要：在邮件服务端和邮件客户端，重复邮件浪费了大量资源。该文提出一种基于 SHA-1 的邮件去重算法，将邮件按大小分开处理，根据 Hash 值快速去除正文相同或相似的重复邮件。实验结果表明了该算法的有效性，其运行速度比传统方法快。
关键词：重复邮件；相似度；去重算法

Email Remove-duplicate Algorithm Based on SHA-1

ZHANG Man, LI Bi-cheng, LIN Chen
(Information Engineering Institute, PLA Information Engineering University, Zhengzhou 450002)

【Abstract】 The duplicate-emails in service terminal and client terminal wastes a lot of recouse. This paper presents an email remove-duplicated algorithm based on Secure Hash Algorithm 1(SHA-1). Based on the size of email, this algorithm detects similarity of emails by comparing sets of Hash value of all paragraphs or all sentences in emails. The experimental results show that this algorithm has a good performance in computing time.
【Key words】 duplicated-email; similarity; remove-duplicate algorithm

1 概述

本文将邮件正文完全相同或近似相同的邮件统称为重复邮件。传统文件去重(去除重复文件)方法在处理重复邮件时存在处理速度慢、对较小或较大的邮件处理性能差的缺点。基于SHA-1(Secure Hash Algorithm 1)的重复邮件检测与去除是近似文本检测技术的一个应用。已有很多对数据挖掘和信息检索领域中近似文本检测技术的研究[1]，Manber提出的sif工具用基于字符串匹配的方法来度量文件相似性，该方法处理速度慢。Garcia-Molina等人提出的SCAM系统借鉴信息检索技术中的向量空间模型，使用基于词频统计的方法来度量文本相似性，该方法计算量大。Heintze开发的KOALA系统采用与sif相似的方法。文献[2]提出shingling方法，计算文章中长为L的shingle指纹，提取特定指纹集并比较 2 个文件指纹集的相似度以查询近似文档，这种方法处理较小文件时性能较差。Chowdhurd提出I-Match算法[3]，通过计算每个词的idf值来选取特征词集，生成一个Hash值，若 2 个文件Hash值相同，则认为它们相似，该方法需要计算大量idf值，且选取的特征词集只是原文件的一部分，查全率和查准率都不高。

2 重复邮件的定义

因为邮件头具有很高的随机伪造性，所以要处理的是经过邮件解析并去除了各种语言的邮件正文文本。本文对重复邮件作如下定义：

定义 1 设 $S(p_i,p_j)$ 是一个判定邮件对 (p_i,p_j) 相似度的测试，且 $0\le S(p_i,p_j)\le 1$。当 $S(p_i,p_j)=1$ 时，邮件 p_i 和邮件 p_j 相同；给定一个阈值 t，如果 $S(p_i,p_j)\ge t$，则邮件 p_i 和邮件 p_j 基于 $S(p_i,p_j)$ 相似，记 $p_i\approx p_j$；把相同和相似的邮件统称为重复邮件。

定义 2 设 h $(h\ge 1)$ 个 SHA-1 指纹的测试 Sh $(p_i,\ p_j)$ 如下：假设邮件 p_i 通过格式分析生成 H 个邮件块 $B_1,\ B_2,\cdots,\ B_H$，若 2 封邮件的 H 相同，则 $Sh\ (p_i,p_j)=h_0/h$，其中，h_0 为邮件 p_i 与邮件 p_j 相同的 SHA-1 指纹个数。给定阈值 t，如果 $Sh\ (p_i,p_j)\ge t$，则称邮件 p_i 和邮件 p_j 重复。

笔者统计了 1 775 封邮件，其中重复邮件为 535 封，大部分重复邮件是垃圾邮件。邮件去重后，一些后续处理(如邮件过滤等)的效率会得到极大提高。

3 邮件去重算法

3.1 SHA-1 算法

SHA-1 算法是Hash算法的一种，它把任意长度的输入通过Hash算法变换成固定长度的输出，该输出就是Hash值。Hash函数具有快速性、单向性和无碰撞性(SHA-1 发生碰撞的概率为 $1/2^{160}$)。目前常用的Hash算法有MD5(Message Digest 5)和SHA-1，SHA-1 算法较成熟，且具有更高的安全性及易实现性[4]，因此，本文采用SHA-1 算法。

SHA-1 算法先对信息报文填补信息位长度：设 M 是原始的报文信息，将它变换为 M_1，如下：

$|M_1|=448 \bmod 512$

if $|M|=448 \bmod 512$ then $|M_1|=|M|+512$

填补内容是：100…00。添加长度信息在末尾的 64 bit，要求原始的报文长度不能超过 2^{64}。

Hash 函数的输入长度通常大于输出长度，是一个压缩过程。处理 512 bit 的报文分组序列(分为 4 组)，其核心包括 4 个循环模块，每个循环由 20 个处理步骤组成，这 4 个循环有相似的地方，但每个循环所用的逻辑函数不同。循环中的 SHA-1 压缩函数为

作者简介：张 曼(1983－)，女，硕士研究生，主研方向：邮件筛选，数据挖掘；李弼程，教授、博士生导师；林 琛，博士研究生
收稿日期：2007-08-12 **E-mail：**yukazmzmzmzmzm@126.com

$$\begin{cases} A = (E + f(t,B,C,D) + S^5(A) + W_t + K_t) \\ B = A, C = S^{30}(B), D = C, E = D \end{cases} \quad (1)$$

其中，A, B, C, D, E 为 5 个缓存器；t 为步数，$0 \le t \le 79$；$f(t,B,C,D)$ 为 t 步的基本逻辑函数；S^k 为循环左移 k 位给定的 32 位字；W_t 为从当前 512 个数据块导出的 32 位字；K_t 为用于加法的常量，有 4 个不同的值。

3.2 邮件去重算法

文献[5]提出 ASD 算法用来去除重复的垃圾邮件，在处理较小邮件时性能较好。它使用的测试数据是 http://www.spamarchive.org 提供的 29 996 封邮件，这些邮件大小都略大于 7 KB。而中科院邮件系统统计发现，实际应用中大于 20 KB 的邮件占 40%左右，这些大于 20 KB 的邮件正文中包含较多句子，分句效率很低，ASD 算法性能会极大降低。因此，本文提出了更适合于较大邮件的去重算法。

3.2.1 邮件解析

邮件是结构性文件，一般包括邮件头、邮件正文和附件 3 个部分。邮件正文分为纯文本和超文本 2 种格式，纯文本邮件可直接提取，而对超文本邮件需要剔除其中所有 HTML 语言的标记符号，还原为纯文本后再提取。

3.2.2 相似度

传统计算文本相似度的决策函数如下：令 $F(a)$ 表示文档 a 的指纹集，$F(b)$ 表示文档 b 的指纹集，$S(a,b)$ 表示文档 a 和文档 b 的相似度，则决策函数为

$$S(a,b) = \frac{|F(a) \cap F(b)|}{|F(a) \cup F(b)|} \quad (2)$$

此相似度决策函数没有考虑各指纹顺序对相似度的影响，如指纹为(1, 2, 3, 4)和指纹为(4, 3, 2, 1)的文档相似度值为 1，但实际上 2 个文档应判定为不重复。本文考虑了指纹的位置因素，对距离近的指纹元素赋以较高权值，对距离远的指纹元素赋以较低权值。若邮件 X 的指纹为 $(x_1, x_2, \cdots, x_m)$，邮件 Y 的指纹为 $(y_1, y_2, \cdots, y_n)$，则 2 封邮件的相似度决策函数为

$$Sim(X,Y) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(\max\{m,n\} - |i-j|) \cdot \sigma_{ij}}{m \cdot n} \quad (3)$$

其中，i，j 分别为 x_i，y_j 的下标，代表各指纹元素的位置；若指纹元素 x_i 和指纹元素 y_j 相同，则 σ_{ij} 为 1，否则 σ_{ij} 为 0；分母 $m \cdot n$ 是 2 个指纹完全相等时的权重。假如出现指纹为(1, 2, 3, 4)和(4, 3, 2, 1)的文档，用式(3)解得其相似度为 0.5，只要设定的阈值超过 0.5 就不会把 2 封邮件判为重复邮件，这种相似度计算方法减少了指纹元素的位置对相似度的影响。

3.2.3 去重算法

一封大小为 20 KB 的邮件，经过邮件解析除去邮件头和附件(据统计邮件头的平均大小为 1 KB 左右，带有附件的邮件很少)，其正文大小为 19 KB，其中一句话约包含 60 B～100 B，取最大值 100 B，则这封邮件正文含 190 个句子。若对每句都进行求 Hash 值、做比较、求相似度等处理，运行时间将呈非线性增长，代价很高。因此，本文提出的邮件去重算法将邮件按大小分开处理。

对小于 20 KB 的邮件，先将正文中形如“e.g”等带句号的缩写词去掉，不去除缩写词会导致句子被断成一些小短句，如“e”，“g”，影响处理性能。去除缩写词后以句号、问号、感叹号等为标志，将正文按句子划分，计算出各句的 Hash 值，放在<1，句子数，各句 Hash 值列表>中。对大于等于 20 KB 的邮件，以回车为划分标志，将邮件正文按段划分，计算出各段 160 bit 的 Hash 值，放在<0，段数，各段 Hash 值列表>中；然后先判断是句子还是段落，再比较段落数或句子数，对句子个数相近的邮件进行相似度计算，超过给定阈值的视为重复邮件。

当读取一封邮件，要判断是否为重复邮件时，具体步骤如下：

(1)获得邮件大小，进行邮件解析，将邮件正文还原为小写纯文本格式；

(2)若邮件大于 20 KB，则转步骤(3)，否则转步骤(4)；

(3)以回车为标志分段，用 SHA-1 计算各段 Hash 值，存储在<0，段数，各段 Hash 值列表>中；

(4)去除缩写词，断句并计算各句的 Hash 值，存储在<1，句数，各句 Hash 值列表>中；

(5)读取数据库中存储的<0(1)，段(句)数，各段 Hash 值列表>，先判断是段还是句，再进行段(句)数比较，数量相近的进行 Hash 值列表相似度计算，超过阈值 t 的，给这封邮件做重复标志，若直到数据库搜索完毕相似度都未超过阈值，则将该邮件加入数据库。

每次进行去重处理前数据库都为空。一般情况下，若阈值设为 1，则只能去除完全相同的邮件，查全率会降低；若阈值设定得太小，就只能去除小部分相同的邮件，查准率会降低。因此，阈值一般设在 0.5～0.9 之间。

运行速度、查准率和查全率可以从不同角度来评价一个算法性能的好坏，但当 2 个算法的查准率和查全率相近时，对算法进行综合比较和性能分析会很困难。本文利用查准率和查全率的调和平均值作为评测算法的综合指标，定义算法的查准率为 p，查全率为 r，则算法性能可以用 γ 表示：

$$\gamma = \frac{2}{1/p + 1/r} \quad (4)$$

4 实验结果

与 Duplicate Email Remover For Outlook 相比，Duplicate Email Remover 只能去除邮件头重复的邮件，比如它会去除相同发送者发给相同接收者的邮件，可能会去除掉很多发送者和接受者相同而内容并不重复的邮件。本文提出的算法是基于邮件正文内容的去重算法，可以去除内容完全相同或只是个别词句有少许差别的邮件，具有更高的实用价值。

4.1 运行时间

本文邮件去重算法的时间复杂度为 $O((160m)^2 N^2)$，其中，m 为 SHA-1 值的个数；N 为邮件个数。与 ASD 算法相比，当邮件较小时，因为本文算法去除了一些缩写词，所以 m 值小于等于 ASD 算法的 m 值；当邮件较大时，本文算法段数小于句子数，因此，其 m 值小于 ASD 算法的 m 值，且时间复杂度低于 ASD 算法。

实验数据是收集到的 1 500 封邮件。编译环境为 Visual C++。电脑配置为 Pentium 4 CPU 2.4 GHz，内存 256 MB，硬盘 80 GB，操作系统为 Window XP。本文算法和 ASD 算法的运行时间如表 1 所示。

表 1 本文算法与 ASD 算法运行时间比较 s

算法	运行时间
本文算法	49.5
ASD 算法	58.8

由表 1 可以看出，本文算法的运算速度略高于 ASD 算法。

4.2 不同相似度阈值下的算法结果

本文提出的基于位置信息的相似度计算方法对查全率和查准率的提高有一定作用。在相似度分别为 0.5, 0.6, 0.7, 0.8, 0.9 下的算法结果如表 2 所示。

表 2 不同相似度阈值下的实验结果

阈值	查准率/(%)	查全率/(%)	γ/(%)
0.5	97.5	100.0	98.7
0.6	98.0	100.0	99.0
0.7	98.3	98.7	98.4
0.8	98.3	98.2	98.3
0.9	98.8	97.3	97.3

由表 2 可以看出，阈值越高算法的查准率越高而查全率越低。考虑 γ 值，可以发现相似度阈值设为 0.6 时算法性能较好。

4.3 不同重复邮件比例下的查准率和查全率

表 3 给出了本文算法和 ASD 算法在不同邮件数和重复邮件比例情况下的查全率和查准率的比较。

表 3 不同重复邮件比例下查准率和查全率

邮件数量	重复邮件	本文查准率/(%)	本文查全率/(%)	本文 γ/(%)	ASD 查准率/(%)	ASD 查全率/(%)	ASD γ/(%)
500	100	97.0	100.0	98.5	97.0	100.0	98.5
500	200	97.5	99.5	97.7	96.5	99.0	98.5
1 000	150	98.0	100.0	98.0	97.3	98.8	99.0
1 000	300	97.6	99.2	98.3	97.5	99.2	98.4
2 000	200	97.0	99.5	97.7	97.0	98.5	98.2
2 000	300	97.8	100.0	97.8	96.2	99.5	98.9

由表 3 中查准率、查全率和 γ 的变化情况可以看出，在邮件数量很大的情况下，本文算法性能依然稳定，重复邮件逐渐增加时，没有发现明显的性能下降现象，对任意比例的重复邮件，查全率都非常的高。从 γ 值可以看出，本文算法有更好的综合性能。

4.4 鲁棒性

对邮件进行修改，添加无关内容到邮件中，测试本文算法的鲁棒性，结果如表 4 所示。

由表 3 可以看出，当修改比例增加且修改位置分散时查准率有下降趋势但都不低于 97%，查全率基本不受影响。这是因为上述对邮件的修改只会改变一个或几个句子(段落)的 Hash 值，只占 Hash 值列表的小部分，只要相似度阈值不要设定得过高，就能体现很好的鲁棒性。

表 4 不同修改比例下本文算法的查准率和查全率

邮件大小/KB	修改位置	修改比例/(%)	查准率/(%)	查全率/(%)
2	集中	1	99.50	99.5
2	集中	5	99.25	100.0
2	集中	20	98.00	99.9
20	集中	1	99.10	99.9
20	集中	5	97.60	100.0
20	集中	20	96.00	99.9
2	分散	1	99.20	100.0
2	分散	5	98.00	100.0
2	分散	20	97.50	99.9
20	分散	1	99.00	99.8
20	分散	5	98.30	100.0
20	分散	20	97.00	100.0

5 结束语

本文提出一种基于 SHA-1 算法的去除重复邮件算法，适用于大量邮件中的邮件去重。重复邮件中多数是垃圾邮件，小部分是服务器对邮件的重复转发，它们浪费了大量存储资源。在服务器端、客户端、过滤、分析处理等方面，邮件去重具有重要意义。

参考文献

[1] 鲍军鹏, 沈钧毅, 刘晓东, 等. 自然语言文档复制检测研究综述[J]. 软件学报, 2003, 14(10): 1753-1760.

[2] Broder A Z, Glassman S C, Manasse M S. Syntactic Clustering of the Web[C]//Proceedings of the 6th International Web Conference. California, USA: [s. n.], 1997.

[3] Chowdhury A, Flexler F, Grossman O, et a1. Collection Statistics for Fast Duplicate Document Detection[J]. ACM Transactions on Information Systems, 2002, 20(2):171-191.

[4] 王建勇, 谢正茂, 雷 鸣, 等. 近似镜像网页检测算法的研究与评测[J]. 电子学报, 2000, 28(增刊): 130-132.

[5] 詹 川, 卢显良, 侯孟书, 等. 基于签名的近似垃圾邮件检测算法[J]. 计算机工程, 2006, 32(5): 122-124.

（上接第 259 页）

器，Web 应用服务器就能直接和分配给它的查询服务器进行 Web Services 调用，获取搜索结果。

4.2 实现方法

本文的调度程序是一个网络监听程序，主控服务器和 Web 应用服务器都通过 Socket 网络接口与其通信。2 个监听命令 set 和 get，分别用于设置查询服务器状态和获取可用查询服务器。set 的使用方式是主控服务器向调度程序通过 socket 发送 set 字符串、状态码及需要设置状态的查询服务器的 IP 地址，调度程序接收到该调用后，就会重新设置相应查询服务器的状态码；get 的使用方法是 Web 应用服务器向调度程序通过 socket 发送 get 字符串，调度程序通过一个调度算法将一个当前可用查询服务器的 IP 地址发送给 Web 应用程序。

使用到的基本调度方法是轮询使用当前可用的查询服务器，每个查询服务器都有一个调用次数变量和一个当前状态变量。每次分配查询服务器时，在当前状态码为可用的查询服务器中选出调用次数最少的那个，作为此次的分配对象。

5 结束语

现有应用系统种类繁多，跨平台、跨编程语言的系统集成技术的作用越来越重要。Web Services 在黄页搜索引擎系统中的成功应用表明它在系统集成方面的重要作用，本文提出的黄页搜索引擎已正式上线(http://www.locoso.com 和 http://www. e118114.cn)，实际测试效果良好，系统运行稳定。Web Services 的安全性及对系统分布式调度策略的完善有待进一步研究。

参考文献

[1] 胡运发. 互关联后继树——一种新型全文数据库数学模型[D]. 上海: 复旦大学计算机与信息技术系, 2002.

[2] 申 展. 互关联后继树模型研究[D]. 上海: 复旦大学计算机与信息技术系, 2004.

[3] 刘 冬. 应用 Axis 开始 Web 服务之旅[EB/OL]. (2003-07-01). http://www. ibm.com/.developerworks/cn/webservices/ws-startAxis/index.html.

[4] Andrew S T, Maarten V S. 分布式系统: 原理与范型[M]. 杨剑峰, 常晓波, 李 敏, 译. 北京: 清华大学出版社, 2004.