(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 101901257 B
(45)授权公告日 2012.07.04

(21)申请号 201010232463.2
(22)申请日 2010.07.21
(73)专利权人 北京理工大学
地址 100081 北京市海淀区中关村南大街 5 号
(72)发明人 嵩天 黎达
(51)Int.Cl.
G06F 17/30(2006.01)
(56)对比文件
CN 101051321 A,2007.10.10, 全文.
CN 101251845 A,2008.08.27, 全文.
CN 1477559 A,2004.02.25, 全文.
Tian Song 等.AN ALGORITHM OF LARGE-SCALE APPROXIMATE MULTIPLE STRING MATCHING.《Department of Computer Science and Technology》.2006, 全文.
审查员 杨春雨

权利要求书 3 页 说明书 10 页

(54)发明名称

一种搜索引擎中的多字符串匹配方法

(57)摘要

本发明涉及一种多字符串匹配方法,属于字符串匹配技术领域。本发明基于传统的 Wu-Manber 方法,对规则集中的长字符串和短字符串进行了分离,在建 SHIFT 表时,对规则集中的长字符串和短字符串采取不同的处理方式,从而使得 SHIFT 表的最大表项不受到短字符串长度的限制,克服了其最大跳跃距离受到规则集中最短字符串长度限制的不足;通过引入 HOT 表,利用匹配过程中查找 HOT 的方法,使得窗口的最大跳跃距离得到增大的同时不会遗漏短字符串。本发明方法具有更高的匹配效率。

CN 101901257 B

1. 一种搜索引擎中的多字符串匹配方法，其特征在于：包括一个预处理阶段和一个扫描阶段；

预处理阶段包括构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT，其具体工作步骤如下：

第 1 步：设置 HOT 表的大小 SUM 以及 HOT 表选取的字符块的长度 s；其中，SUM ≥ 256 并且 SUM 为正整数；s = 2 或 3；

第 2 步：将模式串集合 P 中的全部模式串分为长模式串和短模式串两类，长模式串集合记为 P_1，短模式串集合记为 P_2，短模式串的数量记为 $SumP_2$；具体为：

第 a 步：确定跳越窗口的宽度与最短模式串长度的比值 r，r 为正整数，其取值范围满足公式 (1)；

$$2 \le r \le \frac{l\max}{l\min} \qquad (1)$$

其中，lmax 为模式串集合 P 中最长的模式串的长度；lmin 为模式串集合 P 中最短的模式串长度；

第 b 步：根据公式 (2) 确定分类标准 M 值：

M = r×lmin-(r-1)×s (2)

第 c 步：当模式串的长度大于 M 时，则判断其为长模式串；否则，判断其为短模式串；

第 d 步：判断当前的短模式串的数量 $SumP_2$ 是否满足 $SumP_2 \le 1.5 \times SUM$，如果满足，执行第 3 步；否则，减小 r 值，并确保 r 满足 $2 \le r \le \frac{l\max}{l\min}$，然后返回到第 b 步；

第 3 步：对于全部长模式串的前 M 个字符组成的字符串 StringL 以及全部短模式串的前 lmin 个字符组成的字符串 StringS 进行操作，构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT；具体为：

① HASH 表：HASH 表的每个表项指向所述字符串 StringL 或 StringS 最后 B 个字符被哈希到该表项的模式串，如果有多个模式串被哈希到同一表项，则采用链式存储结构存储；其中，B 为正整数，其值根据实际情况确定；

② PREFIX 表：存储所述字符串 StringL 或 StringS 前 B′ 个字符的哈希值；其中，B′ 为正整数，其值根据实际情况确定；

③ HOT 表：依次对短模式串集合 P_2 中的所有模式串作如下操作：

第 a 步：将指针指向其起始位置，向后取长度为 s 的字符块，计算其哈希值为 h_hot，将 HOT[h_hot] 设置为 1；

第 b 步：将指针后移一位；判断指针与该字符串的结束标识符之间的距离是否为 (s-1)，如果不是，执行第 a 步；否则，结束操作；

经过上述步骤的操作，即可完成 HOT 表的构建；

④ SHIFT 表：

首先，将 SHIFT 表中的所有项赋值为 M-B+1；

然后，对长模式串集合 P_1 中的模式串依次做如下处理：

第 a 步：将指针指向该模式串的第 M 个字符，并用 qi 表示当前指针指向字符串中的位置，qi 为正整数，qi 的初始值为 M；

第 b 步：向前取长度为 B 的字符块，计算其哈希值为 h_shift_1，将 SHIFT[h_shift_1] 的值设置为 M-qi；

第 c 步：将指针向前移动一个字符，并为 qi 赋值为 qi-1；判断距离该模式串的起始字符的距离是否小于 B-1，如果不是，回到第 b 步；否则，结束操作；

再对短模式串集合 P_2 中的模式串依次做如下处理：

第 a 步：将指针指向该模式串的第 lmin 个字符，并用 qj 表示当前指针指向字符串中的位置，qj 为正整数，qj 的初始值为 lmin；

第 b 步：向前取长度为 B 的字符块，计算其哈希值为 h_shift_s，将 SHIFT[h_shift_s] 的值设置为 lmin-qj；

第 c 步：将指针向前移动一个字符，并为 qj 赋值为 qj-1；判断距离该模式串的起始字符的距离是否小于 B-1，如果不是，回到第 b 步；否则，结束操作；

经过上述步骤的操作，即可完成 SHIFT 表的构建；

所述 HASH 表、PREFIX、SHIFT 表和 HOT 表在建立时所用到的哈希函数根据不同情况进行选择；

在扫描阶段，按如下步骤进行：

第 1 步：设一指针 q_text，指向文本 T 的第 M 个字符；

第 2 步：从当前指针往前的 B-1 个字符开始，向后扫描 B 个字符，使用预处理阶段建立 HASH 表所用到的哈希函数，计算该 B 个字符的哈希值 h；

第 3 步：查 SHIFT 表，找到 SHIFT[h]；如果 SHIFT[h] 等于 0，执行第 4 步；否则，跳转到第 7 步；

第 4 步：从当前指针往前的 M-1 个字符开始，向后扫描 B′ 个字符，使用预处理阶段建立 PREFIX 表所用到的哈希函数，计算这 B′ 个字符的前缀哈希值 h_long；从当前指针往前的 lmin-1 个字符开始，向后扫描 B′ 个字符，使用预处理阶段建立 PREFIX 表所用到的哈希函数，计算这 B′ 个字符的前缀哈希值 h_short；

第 5 步：查 HASH 表，找到 HASH[h] 的指针，遍历链表；对链表中的每个模式串，如果它在 PREFIX 表的值与相应的前缀哈希值相等，则将文本 T 和模式串逐一字符进行比较；判断是否完全匹配；如完全匹配，则报告完全匹配位置；否则，不报告；对于长模式串，匹配的起始位置为当前指针位置往前 M-1 个字符处；对于短模式串，匹配的起始位置为当前指针位置往前 lmin-1 个字符处；

第 6 步：将指针 q_text 向后移动一个字符，转到第 8 步；

第 7 步：若 SHIFT[h] 不大于 (lmin-B+1)，则将指针向后移动 SHIFT[h] 个距离；否则，进行如下操作：

第 a 步：设置 r′ ＝ 1，

第 b 步：从当前指针所在位置往后 ((r′ +1)×(lmin-s)-(B-1)) 个字符的位置处向前取长度为 s 的字符串；计算其哈希值 hash_h，判断“HOT[hash_h] ＝ 0”是否成立；若成立，执行第 c 步；否则，跳转到第 d 步；

第 c 步：判断“(r′ +1)×lmin-(r′)×s-(B-1) ＜ SHIFT[h]”是否成立，若成立，将 r′ 取值为 (r′ +1)，返回到第 b 步；否则，将指针向后移动 SHIFT[h] 个字符的距离；

第 d 步：令 dis ＝ ((r′ +1)×lmin-r*s-(B-1))，并将指针向后移动 dis 个字符的距

离；

第 8 步：判断指针 q_text 是否指向文本 T 的结束符，如指向结束符，则结束；否则，转到第 2 步；

经过上述步骤的操作，即可完成多个模式串的匹配。

一种搜索引擎中的多字符串匹配方法

技术领域

[0001] 本发明涉及一种搜索引擎中的多字符串匹配方法，属于字符串匹配技术领域。

背景技术

[0002] 在计算机领域，字符串匹配一直是计算机领域研究的焦点之一。字符串匹配问题可以描述为：已知需要匹配的 t(t 为正整数）个子串（通常称之为模式串，或规则），用 P_1，P_2，…，P_t 表示，需要检索的字符串（通常称之为文本），用 T[1…n](n 为正整数）表示，查找在文本 T[1…n] 中所有出现的模式串，并报告其出现的位置。所谓多模式匹配，就是在文本串 T[1…n] 中一次匹配多个模式串 P_1，P_2，…，P_t，t ＝ 1 时，多模式匹配蜕化为单模式匹配。

[0003] 字符串匹配在拼写检查、语言翻译搜索引擎等应用中起着关键的作用；同时，字符串匹配也是众多信息内容安全系统中的关键技术之一。其中，多字符串匹配的方法目前已经广泛用于网络信息过滤，入侵检测系统和生物信息计算的基因序列比较等实际应用中。

[0004] 这些应用的共同特点有以下两个方面：一是需要处理大量的数据（人类基因组共有 30 多亿个碱基对；2009 年 6 月，中国网络国际出口带宽达到 747541Mbps）；二是需要匹配的关键词条目多（以基因序列为例，关键词条目达到 $O(10^4)$ 的数量级）。随着网络以及生物学的发展，对多字符串匹配方法的处理能力提出了更高的要求。

[0005] 在传统的多字符串匹配方法中，Wu.Sun 和 Udi.Manber 在文献《A Fast.Algorithm for Multi-Pattern Searching》中提出的 Wu-Manber 方法，采用了跳跃不可能匹配的字符策略和 HASH 散列的方法，加速匹配的进行，在许多相关领域中得到了应用。

[0006] Wu-Manber 方法包括一个预处理阶段和一个扫描阶段。

[0007] 在预处理阶段，首先计算模式串集合 P 中最短的模式串长度，记为 m。然后，对所有模式串（仅考虑前 m 个字符组成的模式串）构建哈希表（记为 HASH）、跳转表（记为 SHIFT）和前缀表（记为 PREFIX）。HASH 表的每个表项指向最后 B(B 为正整数，其值根据实验情况择优选择）个字符被哈希到该表项的模式串，如果有多个模式串被哈希到同一表项，则采用链式存储结构存储；SHIFT 表用于在扫描文本串的时候，根据读入字符串决定可以跳过的字符数，其最大值为 (m-B+1)，其最大值也成为跳越窗口的宽度；PREFIX 表存储的是每个模式串前 B′ (B′ 为正整数，其值根据实验情况择优选择）个字符的哈希值。此处，建立 HASH 表和 PREFIX 表所用到的哈希函数根据不同情况进行选择。

[0008] 在扫描阶段，按如下步骤进行：

[0009] 第 1 步：设一指针 q，指向文本 T 的第 m 个字符；

[0010] 第 2 步：从当前指针往前的 B-1 个字符开始，向后扫描 B 个字符，使用预处理阶段建立 HASH 表所用到的哈希函数，计算该 B 个字符的哈希值 h；

[0011] 第 3 步：查 SHIFT 表，找到 SHIFT[h]：如果大于 0，则将指针 q 向后移动 SHIFT[h] 个长度，转到第 2 步；否则转到第 4 步；

[0012] 第 4 步：从当前指针往前的 m-1 个字符开始，向后扫描 B′ 个字符，使用预处理阶

段建立 PREFIX 表所用到的哈希函数，计算这 B′ 个字符的前缀哈希值 h′ ；

[0013] 第 5 步 ：查 HASH 表，找到 HASH[h] 的指针，遍历链表。对链表中的每个模式串，如果它在 PREFIX 表的值与前缀哈希值 h′ 相等，则将文本串和模式串逐一字符进行比较，判断是否完全匹配。如完全匹配，则报告完全匹配位置 q ；否则，不报告 ；

[0014] 第 6 步 ：判断指针 q 是否指向文本串的结束符，如指向结束符，则结束过程 ；否则，将指针 q 向后移动一个字符，转到第 2 步。

[0015] 经过分析与实际运用，发现 WU-MANBER 方法存在以下不足 ：

[0016] SHIFT 表中表项的大小是影响匹配过程中窗口跳跃距离的关键。在传统的 Wu-Manber 方法中 SHFIT 表项中的最大值，同待匹配规则集的最短字符串长度有关，为 (m-B+1)。在实际应用中，有时候规则集中只有少数几个字符串为短字符串，其中绝大多数为长字符串。由于少数的短字符串的存在，使得 SHIFT 表中的值大大减少。这样在匹配的过程中，窗口的跳跃距离也大大减少。在这种情况下，Wu-Manber 方法的性能受到极大影响。特别是长字符串的平均长度远大于少数几个短字符串的平均长度时。例如有一个规则集，其中字符串的最短长度为 100，在对该规则集运用 WU-MANBER 方法进行匹配的时候，SHIFT 表项中的最大值为 (100-B+1)。若往该规则集中加入的少量的短字符串，这些短字符的最短长度为 6，构造成一个新的规则集。根据 Wu-Manber 方法匹配过程，SHIFT 表项中的最大值为 (6-B+1)。若对新的规则集使用 Wu-Manber 方法进行文本的匹配，由于 SHIFT 表项中的最大值将由原来 (100-B+1) 迅速减小到 (6-B+1)，匹配过程中窗口的平均跳跃距离将大大减少，Wu-Manber 方法的快速性将受到影响。

发明内容

[0017] 本发明的目的是克服已有技术存在的不足，提出一种搜索引擎中的多字符串匹配方法。

[0018] 一种搜索引擎中的多字符串匹配方法，包括一个预处理阶段和一个扫描阶段。

[0019] 预处理阶段包括构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT，其具体工作步骤如下 ：

[0020] 第 1 步 ：设置 HOT 表的大小 SUM 以及 HOT 表选取的字符块的长度 s ；其中，SUM ≥ 256 并且 SUM 为正整数 ；s ＝ 2 或 3。

[0021] 第 2 步 ：将模式串集合 P 中的全部模式串分为长模式串和短模式串两类，长模式串集合记为 P_1，短模式串集合记为 P_2，短模式串的数量记为 $SumP_2$ ；具体为 ：

[0022] 第 a 步 ：确定跳越窗口的宽度与最短模式串长度的比值 r，r 为正整数，其取值范围满足公式 (1)。

[0023]

$$2 \le r \le \frac{l\max}{l\min} \qquad (1)$$

[0024] 其中，lmax 为模式串集合 P 中最长的模式串的长度 ；lmin 为模式串集合 P 中最短的模式串长度。

[0025] 第 b 步 ：根据公式 (2) 确定分类标准 M 值 ：

[0026] M ＝ r×lmin-(r-1)×s (2)

[0027] 第 c 步：当模式串的长度大于 M 时，则判断其为长模式串；否则，判断其为短模式串；

[0028] 第 d 步：判断当前的短模式串的数量 $SumP_2$ 是否满足 $SumP_2 \leq 1.5 \times SUM$，如果满足，执行第 3 步；否则，减小 r 值，并确保 r 满足 $2 \leq r \leq \frac{l\max}{l\min}$，然后返回到第 b 步。

[0029] 第 3 步：对于全部长模式串的前 M 个字符组成的字符串 StringL 以及全部短模式串的前 lmin 个字符组成的字符串 StringS 进行操作，构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT；具体为：

[0030] ① HASH 表：HASH 表的每个表项指向所述字符串 StringL 或 StringS 最后 B(B 为正整数，其值根据实际情况确定）个字符被哈希到该表项的模式串，如果有多个模式串被哈希到同一表项，则采用链式存储结构存储。

[0031] ② PREFIX 表：存储所述字符串 StringL 或 StringS 前 B′（B′ 为正整数，其值根据实际情况确定）个字符的哈希值。

[0032] ③ HOT 表：依次对短字符串集合 P_2 中的所有模式串作如下操作：

[0033] 第 a 步：将指针指向其起始位置，向后取长度为 s 的字符块，计算其哈希值为 h_hot，将 HOT[h_hot] 设置为 1；

[0034] 第 b 步：将指针后移一位；判断指针与该字符串的结束标识符之间的距离是否为 (s-1)，如果不是，执行第 a 步；否则，结束操作；

[0035] 经过上述步骤的操作，即可完成 HOT 表的构建。

[0036] ④ SHIFT 表：

[0037] 首先，将 SHIFT 表中的所有项赋值为 M-B+1；

[0038] 然后，对长模式串集合 P_1 中的模式串依次做如下处理：

[0039] 第 a 步：将指针指向该模式串的第 M 个字符，并用 qi(qi 为正整数）表示当前指针指向字符串中的位置，qi 的初始值为 M；

[0040] 第 b 步：向前取长度为 B 的字符块，计算其哈希值为 h_shift_1，将 SHIFT[h_shift_1] 的值设置为 M-qi；

[0041] 第 c 步：将指针向前移动一个字符，并为 qi 赋值为 qi-1；判断距离该模式串的起始字符的距离是否小于 B-1，如果不是，回到第 b 步；否则，结束操作。

[0042] 再对短模式串集合 P_2 中的模式串依次做如下处理：

[0043] 第 a 步：将指针指向该模式串的第 lmin 个字符，并用 qj(qj 为正整数）表示当前指针指向字符串中的位置，qj 的初始值为 lmin；

[0044] 第 b 步：向前取长度为 B 的字符块，计算其哈希值为 h_shift_s，将 SHIFT[h_shift_s] 的值设置为 lmin-qj；

[0045] 第 c 步：将指针向前移动一个字符，并为 qj 赋值为 qj-1；判断距离该模式串的起始字符的距离是否小于 B-1，如果不是，回到第 b 步；否则，结束操作。

[0046] 经过上述步骤的操作，即可完成 SHIFT 表的构建。

[0047] 所述 HASH 表、PREFIX、SHIFT 表和 HOT 表在建立时所用到的哈希函数根据不同情况进行选择。

[0048] 在扫描阶段，按如下步骤进行：

[0049] 第 1 步 :设一指针 q_text,指向文本 T 的第 M 个字符 ;

[0050] 第 2 步 :从当前指针往前的 B-1 个字符开始,向后扫描 B 个字符,使用预处理阶段建立 HASH 表所用到的哈希函数,计算该 B 个字符的哈希值 h ;

[0051] 第 3 步 :查 SHIFT 表,找到 SHIFT[h] ;如果 SHIFT[h] 等于 0,执行第 4 步 ;否则,跳转到第 7 步 ;

[0052] 第 4 步 :从当前指针往前的 M-1 个字符开始,向后扫描 B′ 个字符,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_long ;从当前指针往前的 lmin-1 个字符开始,向后扫描 B′ 个字符,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_short ;

[0053] 第 5 步 :查 HASH 表,找到 HASH[h] 的指针,遍历链表。对链表中的每个模式串,如果它在 PREFIX 表的值与相应的前缀哈希值 (对于长模式串,为 h_long,对于短模式串,为 h_short) 相等,则将文本 T 和模式串逐一字符进行比较 (对于长模式串,匹配的起始位置为当前指针位置往前 M-1 个字符处 ;对于短模式串,匹配的起始位置为当前指针位置往前 lmin-1 个字符处),判断是否完全匹配。如完全匹配,则报告完全匹配位置 ;否则,不报告 ;

[0054] 第 6 步 :将指针 q_text 向后移动一个字符,转到第 8 步。

[0055] 第 7 步 :若 SHIFT[h] 不大于 (lmin-B+1),则将指针向后移动 SHIFT[h] 个距离 ;否则,进行如下操作 :

[0056] 第 a 步 :设置 r′ = 1,

[0057] 第 b 步 :从当前指针所在位置往后 ((r′ + 1)×(lmin-s)-(B-1)) 个字符的位置处向前取长度为 s 的字符串 ;计算其哈希值 hash_h,判断“HOT[hash_h] = 0”是否成立 ;若成立,执行第 c 步 ;否则,跳转到第 d 步 ;

[0058] 第 c 步 :判断“(r′ +1)×lmin-(r′)×s-(B-1) < SHIFT[h]”是否成立,若成立,将 r′ 取值为 (r′ +1),返回到第 b 步 ;否则,将指针向后移动 SHIFT[h] 个字符的距离 ;

[0059] 第 d 步 :令 dis = ((r′ +1)×lmin-r*s-(B-1)),并将指针向后移动 dis 个字符的距离。

[0060] 第 8 步 :判断指针 q_text 是否指向文本 T 的结束符,如指向结束符,则结束 ;否则,转到第 2 步。

[0061] 经过上述步骤的操作,即可完成多个模式串的匹配。

[0062] 有益效果

[0063] 本发明的一种搜索引擎中的多字符串匹配方法与已有技术比较,具有以下优点 :本发明结合了原有 Wu-Manber 方法的窗口跳跃的优点,克服了其最大跳跃距离受到规则集中最短字符串长度限止的不足。通过引入 HOT 表,利用匹配过程中查找 HOT 的方法,使得窗口的最大跳跃距离得到增大的同时不会遗漏短字符串。本发明方法具有更高的匹配效率。

具体实施方式

[0064] 下面结合附图和具体实施例对本发明方案进行详细说明。

[0065] 实施例中,待匹配的模式串集合 P 中有 3 个模式串,分别为“english”、“kilometer”、“fine”,对应长度为 7、9、4,其编号分别为 0,1,2。其文本 T 的内容为“vmogenglishsdyfine”;使用本发明提出的方法在文本 T 中搜索模式串“english”、

“kilometer”、“fine”的具体过程如下：

[0066] 预处理阶段包括构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT，其具体工作步骤如下：

[0067] 第 1 步：设置 HOT 表的大小 SUM 以及 HOT 表选取的字符块的长度 s；将 SUM 设置为 256，取 s = 2。

[0068] 第 2 步：将模式串集合 P 中的全部模式串分为长模式串和短模式串两类，长模式串集合记为 P_1，短模式串集合记为 P_2，短模式串的数量记为 $SumP_2$；具体为：

[0069] 第 a 步：确定跳越窗口的宽度与最短模式串长度的比值 r，在本例中 lmax = 9，lmin = 4，由公式 (1) 可得 r = 2。

[0070] 第 b 步：根据公式 (2) 确定分类标准 M 值：M = 2×4-(2-1)×2 = 6。

[0071] 第 c 步：当模式串的长度大于 6 时，则判断其为长模式串；否则，判断其为短模式串；因此将“english”、“kilometer”归入长模式串集合；“fine”归入短模式串集合。该过程结束后，将原有模式串集合划分为长模式串集合与短模式串集合；长模式串集合包括：“english”、“kilometer”；短模式串集合包括：“fine”。

[0072] 第 d 步：判断当前的短模式串的数量 $SumP_2$ 是否满足 $SumP_2 \leqslant R \times SUM$，此时取 R = 50%，由于 $SumP_2 = 1$，$R \times SUM = 50\% \times 256 = 128$，满足 $SumP_2 \leqslant R \times SUM$，执行第 3 步。

[0073] 第 3 步：对于全部长模式串的前 6 个长度的字符进行截取，对短模式串的前 4 个字符进行截取，得到 StringL 与 StringS 的集合，如表 1 所示。

[0074] 表 1StringL 与 StringS 集合

[0075]

StringL	e	n	g	l	i	s
	k	i	l	o	m	e

[0076]

StringS	f	i	n	e

[0077] 根据 StringL 与 StringS，构建哈希表 HASH、跳转表 SHIFT、前缀表 PREFIX 和短模式串过滤表 HOT；

[0078] 首先选定构造时所用到的哈希函数，对于 HASH 表，PREFIX 表，以及 SHIFT 表的构造，均选用哈希函数 hash1；对于 HOT 表，选用哈希函数 hash2；两个函数的定义如下：

[0079]

```
unsigned int hash1(char * str){
    unsigned int hash_value = 0;
    while( *str!= NULL){
        hash_value <<= 6;
        hash_value += *str++;
    }
    return hash_value & 0x2ffff;
}
unsigned int hash2(char * str){
    unsigned int hash_value = 0;
    while( *str!= NULL){
        hash_value <<= 6;
        hash_value += *str++;
    }
    return hash_value & 0xff;
}
```

[0080] 将 HASH 表、SHIFT 表的大小选取为 0x2ffff，PREFIX 表的大小为规则集中模式串的数目 3，HOT 表的大小如前文中所示，用 16 进制表示为 0xff。

[0081] 具体构造过程为：

[0082] ① HASH 表：选取 B = 2，对于 StringL 集合中的“englis”，选取“is”，计算哈希值为 6835，将“english”添加至 HASH[6835] 中；在“kilome”的末尾，选取“me”，计算其哈希值为 7077，将“kilometer”添加至 HASH[7077] 中；对于 StringS 集合中的“fine”，选取末尾“ne”，计算其哈希值，假定为 7141，将“fine”添加至 HASH[7141] 中。HASH 表如表 2 所示。

[0083] 表 2 本发明方法得到的 HASH 表

[0084]

哈希值	……	6835	……	7077	……	7141	……

对应模式串	NULL	english	NULL	kilometer	NULL	fine	NULL

[0085] ②PREFIX表：选取B′ ＝2，对于StringL集合中的“englis”的起始，选取“en”，计算哈希值为6574，将PREFIX表中“englis”的对应项PREFIX[0]赋值为6574；在“kilome”的起始，选取“ki”，计算其哈希值为6953，将PREFIX表中“kilome”的对应项PREFIX[1]赋值为6953；对于StringS集合中的“fine”，在起始处选取“fi”，计算其哈希值为6633，将PREFIX表中“fine”的对应项PREFIX[2]赋值为6633。PREFIX表如表3所示。

[0086] 表3本发明方法得到的PREFIX表

[0087]

规则编号	0	1	2
PREFIX	6574	6953	6633

[0088] ③HOT表：设定构建HOT表的字符块长度为s＝2，HOT表的大小为256，并将HOT表中的值均初始化为0。对于短模式串集合中的“fine”，首先将指针指向其起始位置，向后选取长度为2的字符块“fi”，计算其哈希值为233，则令HOT[233]＝1；将指针往后移动一位，向后选取长度为2的字符块“in”，计算其哈希值为174，令HOT[174]＝1；将指针往后移动一位，向后选取长度为2的字符块“ne”，计算其哈希值为229，则令HOT[229]＝1；至此，fine中任意长度为2字符块均进行了HOT表的填写处理，对“fine”的处理结束。对短模式集合P_2中的所有模式串，进行相同的处理。本例中，P_2中仅有一个模式串，HOT表的构造结束。HOT表如表4所示。

[0089] 表4本发明方法得到的HOT表

[0090]

字符块	in	ne	fi	其他
哈希值	174	229	233	……
HOT	1	1	1	0

[0091] ④SHIFT表：

[0092] 首先，将SHIFT表中的所有项赋值为M-B+1，此时有M＝6，B＝2，即SHIFT表中的所有项初始化为5；

[0093] 然后，对长模式串集合P_1中的模式串进行处理。

[0094] 对于“english”，将指针指向该模式串的第6个字符“s”，此时当前指针的位置为qi＝6。向前取长度为2的字符块“is”，计算其哈希值为6835，将SHIFT[6835]的值设置为M-qi＝0；将指针向前移动一个字符，将qi赋值为qi-1＝5，此时向前取字符块“li”，计算其哈希值为7017，将SHIFT[7017]的值设置和M-qi＝1；将指针继续向前移动一个字符，将qi赋值为qi-1＝4，此时向前取字符块“gl”计算其哈希值为6700，将SHIFT[6700]的值设置为M-qi＝2；继续按照该方法填写SHFTT表，直至指针移至距离模式串起始距离为2，此时qi＝2，向前取字符块“en”，计算其哈希值为6574，将SHIFT[6574]的值设置为

M-qi = 4。再将指针向前移动一个字符，则有 qi = 1，此时距离该模式串的起始字符的距离为 qi-1 = 0 小于 B-1 = 1。至此，对“english”的处理结束。对长模式串集合 P_1 中的另一模式串“kilome”采用相同的方法处理，直至处理完 P_1 中所有模式串。

[0095] 再对短模式串集合 P_2 中的模式串进行处理。

[0096] 对于“fine”，将指针指向该模式串的第 4 个字符“e”，此时当前指针的位置为 qj = 6。向前取长度为 2 的字符块“ne”，计算其哈希值为 7141，将 SHIFT[7141] 的值设置为 lmin-qj = 0 ；将指针向前移动一个字符，将 qj 赋值为 qj-1 = 3，此时向前取字符块“in”，计算其哈希值为 6830，将 SHIFT[6830] 的值设置为 lmin-qj = 1 ；将指针继续向前移动一个字符，将 qj 赋值为 qj-1 = 2，向前取字符块“fi”，计算其哈希值为 6633，将 SHIFT[6633] 的值设置为 lmin-qj = 2。再将指针向前移动一个字符，则有 qj = 1，此时距离该模式串的起始字符的距离为 qj-1 = 0 小于 B-1 = 1。至此，对“fine”的处理结束。由于 P_2 中仅含有模式串“fine”，对 P_2 的处理也结束。从而得到构建好的 SHIFT 表，如表 5 所示。

[0097] 表 5 本发明方法得到的 SHIFT 表

[0098]

字符块	is	me	ne	li	om	in	gl
哈希值	6835	7077	7141	7017	7213	6830	6700
SHIFT	0	0	0	1	1	1	2
字符块	lo	fi	ng	il	en	ki	其他
哈希值	7023	6633	7143	6828	6574	6953	……
SHIFT	2	2	3	3	4	4	5

[0099] 下面对文本 T 进行扫描，寻找模式串，详细过程如下 ：

[0100] 第 1 步 ：设一指针 q_text，指向文本 T 的第 6 个字符“n”。

[0101] 第 2 步 ：从当前指针往前的 1 个字符开始，向后取长度为 2 的字符块“en”，使用预处理阶段建立 HASH 表所用到的哈希函数，计算其哈希值为 6574。

[0102] 第 3 步 ：根据该哈希值，查找 SHIFT 表，可得 SHIFT[6574] = 4 ；转到第 7 步。

[0103] 第 7 步 ：若 SHIFT[h] 大于 3，需要查找 HOT 表。取 r′ = 1，在文本 T 中距离当前指针位置（第 6 个字符）往后 3 个字符（第 9 个字符）处，向前读取长度为 s = 2 的字符块“li”，计算其哈希值并查找 HOT 表，由于 HOT 表中相应项为 0，将 r′ 增大为 2，判断 (r′ +1)×lmin-(r′)×s-(B-1) < 4 是否成立，不成立，因此，可将指针向后移动 SHIFT[6574] = 4 个距离，此时指针指向文本 T 中第 10 个字符。

[0104] 第 8 步 ：判断指针 q_text 未指向文本 T 的结束符，转到第 2 步。

[0105] 第 2 步 ：从当前指针往前的 1 个字符开始，向后取长度为 2 的字符块“is”，使用预处理阶段建立 HASH 表所用到的哈希函数，计算其哈希值为 6835。

[0106] 第 3 步 ：根据该哈希值，查找 SHIFT 表，可得 SHIFT[6574] = 0 ；执行第 4 步。

[0107] 第 4 步 :从当前指针往前的 M-1 ＝ 5 个字符开始,向后扫描 B′ ＝ 2 个字符,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_long ＝ 6574 ;从当前指针往前的 lmin-1 ＝ 3 个字符开始,向后扫描 B′ ＝ 2 个字符,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_short ＝ 6700 ;

[0108] 第 5 步 :查 HASH 表,找到 HASH[6835] 所对应的模式串为“english”,将其在 PREFIX 表中的值“6574”与 h_long 对比,发现相等。因此,将文本与模式串“english”逐一比较,发现完全匹配。于是,报告“english”在文本中的位置。此时,指针 q_text 指向第 11 个字符)

[0109] 第 6 步 :将指针 q_text 向后移动一个字符 (第 12 个字符),转到第 8 步。

[0110] 第 8 步 :判断指针 q_text 未指向文本 T 的结束符,转到第 2 步。

[0111] 第 2 步 :从当前指针往前的 1 个字符开始,向后取长度为 2 的字符块“hs”,使用预处理阶段建立 HASH 表所用到的哈希函数,计算其哈希值为 6771。

[0112] 第 3 步 :根据该哈希值,查找 SHIFT 表,可得 SHIFT[6771] ＝ 5 ;转到第 7 步。

[0113] 第 7 步 :若 SHIFT[h] 大于 3,需要查找 HOT 表。取 r′ ＝ 1,在文本 T 中距离当前指针位置 (第 12 个字符) 往后 3 个字符 (第 15 个字符) 处,向前读取长度为 s ＝ 2 的字符块“yf”,计算其哈希值并查找 HOT 表,由于 HOT 表中相应项为 0,将 r′ 增大为 2,判断 (r′ +1)×lmin-(r′)×s-(B-1) ＜ 5 是否成立,不成立,因此,可将指针向后移动 SHIFT[6574] ＝ 5 个距离,此时指针指向文本 T 中第 17 个字符。

[0114] 第 8 步 :判断指针 q_text 未指向文本 T 的结束符,转到第 2 步。

[0115] 第 2 步 :从当前指针往前的 1 个字符开始,向后取长度为 2 的字符块“in”,使用预处理阶段建立 HASH 表所用到的哈希函数,计算其哈希值为 6830。

[0116] 第 3 步 :根据该哈希值,查找 SHIFT 表,可得 SHIFT[6830] ＝ 1 ;执行第 7 步。

[0117] 第 7 步 :若 SHIFT[h] 不大于 3,则将指针向后移动 1 个距离,此时,指针指向文本 T 中第 18 个字符。

[0118] 第 8 步 :判断指针 q_text 未指向文本 T 的结束符,转到第 2 步。

[0119] 第 2 步 :从当前指针往前的 1 个字符开始,向后取长度为 2 的字符块“ne”,使用预处理阶段建立 HASH 表所用到的哈希函数,计算其哈希值为 7141。

[0120] 第 3 步 :根据该哈希值,查找 SHIFT 表,可得 SHIFT[7141] ＝ 0 ;执行第 4 步。

[0121] 第 4 步 :从当前指针往前的 M-1 ＝ 5 个字符开始,向后扫描 B′ ＝ 2 个字符“dy”,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_long ＝ 6521 ;从当前指针往前的 lmin-1 ＝ 3 个字符开始,向后扫描 B′ ＝ 2 个字符“fi”,使用预处理阶段建立 PREFIX 表所用到的哈希函数,计算这 B′ 个字符的前缀哈希值 h_short ＝ 6633 ;

[0122] 第 5 步 :查 HASH 表,找到 HASH[7141] 所对应的模式串为“fine”,将其在 PREFIX 表中的值“6633”与 h_short 对比,发现相等。因此,将文本与模式串“fine”逐一比较,发现完全匹配。于是,报告“fine”在文本中的位置。此时,指针 q_text 指向第 18 个字符)

[0123] 第 6 步 :将指针 q_text 向后移动一个字符 (第 19 个字符),转到第 8 步。

[0124] 第 8 步 :判断指针 q_text 指向文本 T 的结束符,结束操作。

[0125] 为说明本发明的效果,使用原始 WU-MANBER 方法对本实施例中的数据进行操作,

得到 SHIFT 表如 6 所示：

[0126] 表 6 原始 WU-MANBER 方法得到的 SHIFT 表

[0127]

字符块	gl	lo	ne	ng	il	in	en	ki	fi	其它
SHIFT	0	0	0	1	1	1	2	2	2	3

[0128] 通过两种方法的比较，可以看出：本发明方法将原始的 WU-MANBER 方法中的最大跳跃距离 lmin-1 ＝ 3 扩大至 M-B+1 ＝ 5，使得在匹配的过程中能够跳跃更远的距离，减少跳跃的次数，从而提高匹配的效率。实际上，根据模式串集合构造的 SHFIT 表中值的平均大小能够反映匹配过程中跳跃的平均长度。

[0129] 以上所述仅是本发明的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本发明原理的前提下，还可以做出若干改进，或者对其中部分技术特征进行等同替换，这些改进和替换也应视为本发明的保护范围。