第32卷 第23期 计 算 机 工 程 2006年12月
Vol.32 No.23 Computer Engineering December 2006
·软件技术与数据库· 文章编号：1000—3428(2006)23—0078—04 文献标识码：A 中图分类号：TP301.6

对BM模式匹配算法的一个改进

渠 瑜，王亚弟，韩继红，赵 宇
(解放军信息工程大学电子技术学院，郑州 450004)

摘 要：在分析BM算法的基础上，提出了一个改进的模式匹配算法QBM算法。该算法采用最长前缀的思想，在匹配后缀的时候采用一个位置移动表shift表代替BM算法中的两个移动表，提高了算法的运行效率。从理论和实践两个方面证明了该算法要优于BM算法。
关键词：模式匹配；最长前缀；移动表

Improvement of BM Algorithm for Pattern-matching

QU Yu, WANG Yadi, HAN Jihong, ZHAO Yu
(Institute of Electronic Technology, PLA Information Engineering University, Zhengzhou 450004)

【Abstract】 Based on the analysis of BM algorithm, this article represents a new pattern-matching algorithm, namely QBM algorithm. This algorithm uses the concept of max-length prefix, and applies only one shift table to calculate the movement distance of pattern strand. The article proves that QBM algorithm is superior to BM algorithm from the aspects of theory and practice.
【Key words】 Pattern-matching; Max-length prefix; Shift table

在拼写检查、基于字典的语言翻译、WWW搜索引擎、计算机病毒特征码匹配、数据压缩以及DNA序列匹配等大量应用中都需要使用字符串匹配技术。因此，在计算机科学领域中，串匹配问题一直是研究的焦点之一。在基于分布式WLAN入侵检测系统的规则匹配中，由于绝大多数入侵规则都是以字符串形式存在的，因此研究一个高效实用的字符串匹配算法是非常有必要的。近年来，对字符串匹配问题的研究引起了众多学者和研究人员的广泛关注：1970年，S. A. Cook从理论上证明了一维模式匹配问题可以在O(m＋n)时间内解决(n、m分别为正文和模式的长度)，为串匹配算法的进一步发展奠定了坚实的理论基础；D. E. Knuth，V. R. Pratt和T. H. Morris仿照Cook的证明构造了KMP算法[1]；R. S. Boyer和J. S. Moore设计了BM算法[2]；Karp和Rabin给出了RK算法[3]和随机算法[4]；Vishkin提出了决定性抽样算法[5]。上述5种串匹配算法都是比较精确的，但同时也存在不足。例如，对于KMP算法，其程序设计使用了递归函数来降低时间复杂度，但它在设计思想和算法预处理方面比较复杂，并且较难理解。Vishkin的决定性抽样算法的时间复杂度为O(m＋nlogn)[6]，黄占友的KMP改进算法只适用于特殊字符串[7]等。在实际使用中，BM算法设计简单，所以应用最为广泛。

本文在对BM算法进行了深入分析的基础上，提出了一个改进的模式匹配算法——QBM算法。

1 BM算法简介

1.1 基本概念

记模式串为$X=x_0x_1x_2\cdots x_{m-1}$，长度为m；文本串为$Y=y_0y_1y_2\cdots y_{n-1}$，长度为n；字母表为∑，表空间大小为σ。

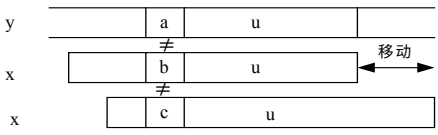
将文本串与模式串中对应的各个字符进行比较，当$x_0x_1x_2\ldots x_{m-1}$与$y_iy_{i+1}y_{i+2}\cdots y_{i+m-1}$对齐时，称为模式串对文本串在位置i的一次尝试。其中，i称为尝试位置。

1.2 基本思想

BM算法在匹配的过程中采用了从后向前对模式串后缀进行比较的策略。在完成一次尝试(包括匹配失败或成功)后，利用预处理好的坏字符移动表与好后缀移动表来确定模式串的后移距离。具体分析如下：

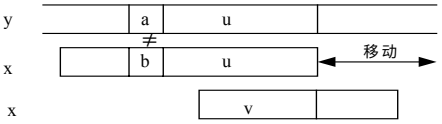
对于模式串X在文本串Y中位置i处的尝试，若前m–j–1次比较已成功完成，而在第m–j次比较时，模式串中的字符$x_j=b$与文本中的字符$y_{i+j}=a$不相同，即有$y_{i+j+1}y_{i+j+2}\cdots y_{i+m-1}=x_{j+1}x_{j+2}\ldots x_{m-1}=u$及$y_{i+j}\neq x_j$，则此时BM算法将根据以下两种情况计算模式串向后移动的距离：

(1)好后缀移动。首先在模式串X中从后往前查找前导字符不是b的u串，如图1所示；如果模式串X中不存在这样的u串，则在文本串Y中u的后缀中查找与X的前缀相同的最长子串v，如图2所示。



(x里还包含u，且该u前时一个不同于b的字符)

图1 好后缀移动



(好后缀移动，x里只含有u的一个后缀)

图2 好后缀移动

作者简介：渠 瑜(1981－)，男，硕士生，主研方向：计算机网络安全，信息系统安全；王亚弟，教授、博导；韩继红，副教授；赵 宇，硕士生
收稿日期：2005-12-22 **E-mail：**a_shu126@hotmail.com

(2)坏字符移动。在模式串X的子串 $x_0x_1x_2...x_{m-2}$ 中从后往前查找文本字符y_{i+j}的第一次出现(如图 3 所示)；如果该子串中不存在字符 y_{i+j}，则显然模式串 X 只可能在 $y_{i+j+1}y_{+j+2}\cdots y_{n-1}$ 确定的位置区间中存在(如图 4 所示)。

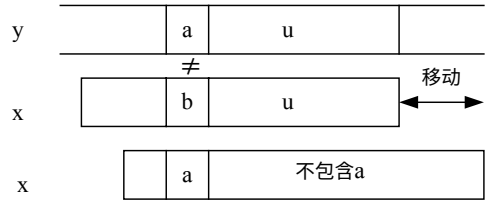


图 3 坏字符移动，x 包含 a

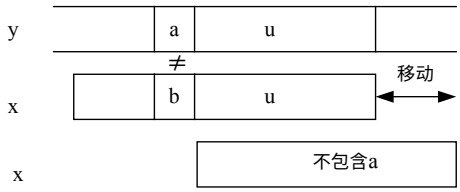


图 4 坏字符移动，x 不包含 a

BM 算法分别根据上述好后缀移动方式与坏字符移动方式计算所得到的移动值创建好后缀移动表和坏字符移动表。在算法运行时，通过查找这两个移动表，并对所得结果进行比较，用移动距离的较大值来向后移动模式串 X 的尝试位置。虽然这种算法在字母表(相对于模式长度)空间较大的情况下非常有效，但由于没有考虑已匹配的后缀及导致匹配失败的当前字符之间的相邻关系，使得算法在总体上来说效率不高。

2 改进的 BM 算法——QBM 算法

虽然 BM 算法在性能上比直接使用 C 库函数 strstr()进行匹配有很大提高，同时也比 KMP 算法要快 3~5 倍，但随着计算机网络数据传输速率的不断提高，必须研究出新的更有效的模式匹配算法，才能保证 WLAN 入侵检测系统的实时性和实用性。

本文在分析了 BM 算法的基础上对其进行改进，提出了一种新的算法——QBM 算法，其基本思想是：在长为 m 的模式串 P 中找出一个最长的前缀子串 subp，设长度为 k，令该子串的末字符为模式串 P 所有字符中最后一个首次出现的字符；在模式匹配过程中，仅当 subp 匹配成功时才对模式 P 余下的部分 endp=P−subp 进行匹配，若 endp 匹配不成功，则模式串 P 从正文中滑过长度不小于 k 的一个字段(endp 在第 sum 个字符时匹配失败，sum≤m−k)；在匹配前缀子串 subp 时，算法采用了一个位置移动表 shift[k][σ]来代替 BM 算法中的好后缀移动表和坏字符移动表(k 是模式串的长度，σ 是每个状态可能的输入字母表∑的大小)，通过位置移动表计算出来的移动值来向后移动模式串 P，以完成匹配过程。算法的具体描述如下：

2.1 基本概念

设∑为有限字符集，对于给定的长为 m 的模式串

$P=p_1p_2\cdots p_m$ ($p_j\in\Sigma$, j=1,2,… ,m)，

定义函数 h: ∑→{1,2,…,m},使得

$$h(p_j)=\begin{cases} i & \text{字符}p_i\text{在模式串P中第1次出现} \\ j & \text{否则} \end{cases},$$

其中，j 为 p_j 在 P 中第 1 次出现的位置；

令 $k=\min\{i \mid h(p_i)=\max\{h(p_j), j=1,2,\ldots,m\}\}$；

subp= $p_1p_2\cdots p_k$；

endp=P−subp= $p_{k+1}p_{k+2}\cdots p_m$；

shift[k][σ]为一个位置移动表；

模式串 P 的周期(period)为 r；

文本串 Y= $y_0y_1y_2\cdots y_{n-1}$，长度为 n。

2.2 QBM 算法基本步骤

(1)预处理阶段

预处理阶段分为两部分：(1)针对模式串 P 求出其最长前缀 subp；(2)根据前缀串 subp 初始化位置移动表——shift 表。

1)计算模式 P 的最长前缀子串 subp 的算法如下：

```
FOR  x ∈∑   DO
h(x) = 1；
DONE
h(P[1]) = m+1；
k= m+1；
FOR  i 从 2 到 m  DO
  IF   h(P[i])=1  THEN
    k=i；
      h(P[i]) = i；
      ENDIF
 DONE
IF  k = m+1  THEN
  k= 1；
 ENDIF
FOR  i 从 1 到 k  DO
  subp[i] = P[i]；
 DONE
FOR i 从 1 到 m−k  DO
  endp[i]= p[k+i]；
 DONE
```

2)初始化位置移动表——shift 表的算法如下：

```
将 shift 表的所有单元设置为 k；
将链表 last 置为空；
FOR  j 从 k−1 到 0 DO
  FOR 链表 last 中所有元素 i DO
     IF x[j] = x[j] THEN
       将元素 i 减 1；
     ELSE IF shift[i][x[j]] = m THEN
       shift[i][x[j]] = i-j；
     ENDIF
DONE
将元素 k-1 插入到 last 头部；
DONE
遍历 last，将其中最小元素值赋给 period；
将链表 last 元素倒转(按从小到大排序)；
g = 0；
WHILE last 中还有元素  DO
     从 last 表头取一元素 h；
     FOR k 从 g 到 h，所有字母 a DO
        IF shift[k][a] = k THEN
          shift[k][a] = i+1；
        ENDIF
     DONE
     g = h+1；
```

DONE

(2)全文扫描阶段

全文扫描过程分为最长前缀匹配和后缀匹配两部分。在尝试位置 i 对模式 P 的最长前缀进行匹配，若在第 j 个位置匹配失败，则将模式 P 从左向右移动 shift[j][y[i+j]]。在最长前缀匹配成功时，在尝试位置 i 对模式 P 的后缀进行匹配，若成功，则模式 P 移动 r，若不成功，则移动 k。

算法的具体流程如图 5 所示。

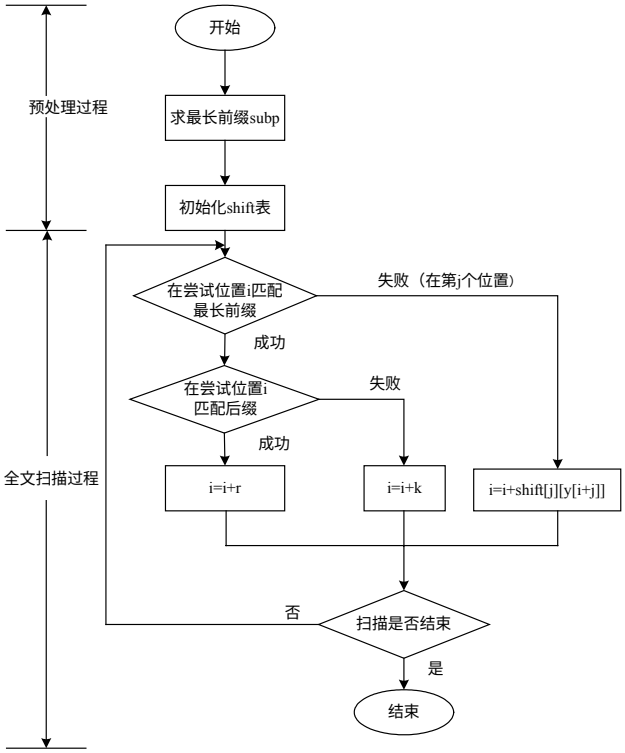


图 5 QBM 算法流程

全文扫描部分核心算法如下：

```
FOR   i 从 0 到 n-m   DO
  FOR   j 从 k-1 到 0，且 y[i+j] ==x[j]   DO
    /*在尝试位置对最长前缀串进行匹配*/
DONE
 IF     j < 0   THEN
        /*最长前缀串匹配成功*/
FOR   j 从 k 到 m-1，且 y[i+j]==x[j]   DO
  /*在尝试位置对后缀串进行匹配*/
DONE
ENDIF
IF   j >= m    THEN
/*后缀串进行匹配成功*/
  i+=period；
ELSE
  /*后缀串进行匹配不成功*/
  i+=k；
 ENDIF
 ELSE
  /*最长前缀串匹配不成功*/
  i+=shift[j][y[i+j]]；
        ENDIF
DONE
```

2.3 QBM 算法复杂度分析

对 QBM 算法复杂度的分析如下：

(1)当在尝试位置对模式串 P 的最长前缀匹配成功时，算法以 k 为步长对正文进行扫描；

(2)当在尝试位置对模式串 P 的最长前缀匹配不成功时，算法以 shift[j][y[i+j]]为步长对正文进行扫描。

BM 算法的复杂度为 O(mn)，设 p 为最长前缀匹配成功的概率，可得 QBM 算法的时间复杂度为 O(pmn/k+mn(1-p))，

$$\frac{pmn}{k}+mn(1-p)=mn\frac{k-(k-1)p}{k}$$

由 k≥1 可得

$$mn\frac{k-(k-1)p}{k}\le mn$$

其中，当 k＝1 或 p＝0 时取等号。当 k＝1 时，此时模式串 P 步长为 1，由同一个字符组成。从统计学的角度来看，这种情况在串模式匹配中非常少见，概率趋近于 0；当 p＝0 时，此时前缀匹配不成功，模式 P 以 shift 中的移动值为步长对全文进行扫描，要优于 BM 算法。因此，就复杂度而言，QBM 算法从整体上优于 BM 算法。

2.4 QBM 算法的改进之处

(1)QBM 算法在对最长前缀进行匹配时，只查找一个表即可得到模式串的移动值；而 BM 算法在匹配过程中需要查找两个表并对这两个查找值进行比较取其中的较大值。因此，每次在向后移动模式串时，QBM 算法都要比 BM 算法少查一次表，同时少做一次比较。

(2)QBM 算法的匹配最长前缀时计算出来的移动值不小于 BM 算法的移动值，从而减少了匹配时间。

2.5 实验结果及分析

评价一个串匹配算法优劣程度的两个重要指标是：算法的运行时间和字符的匹配次数。本节对 QBM、KMP、BM 和 MS(Maximal Shift algorithm)4 种串匹配算法的运行时间和匹配次数进行了实验统计。

实验中选用的文本串出自伍庭芳的著作《America Through the Spectacles of an Oriental Diplomat》中的 1 556 个小写字母，模式串采用在该文中选取长度不同的字符序列 P_1、P_2、P_3、P_4和 P_5 5 个字符串，长度分别为 12、35、64、79 和 99。

在实验中对 4 个串匹配算法进行测试比较，测试程序由 Java 实现，运行结果是 4 个算法的扫描时间和匹配次数。具体实验结果如图 6、图 7 和图 8 所示。

算法	运行时间（ms）		匹配次数	
	总运行时间	每字符运行时间	总匹配次数	每字符匹配次数
KMP	3.04	2.02E-3	602	0.39
BM	1.17	7.52E-4	207	0.13
MS	1.14	7.33E-4	202	0.19
QBM	0.94	6.04E-4	167	0.11

图 6 实验结果对比

其中，在图 6 中，每字符运行时间等于总运行时间除以文本长度，即文本中每个字符的扫描时间；每字符匹配次数等于总匹配次数除以文本长度，即文本中每个字符的平均匹配次数。

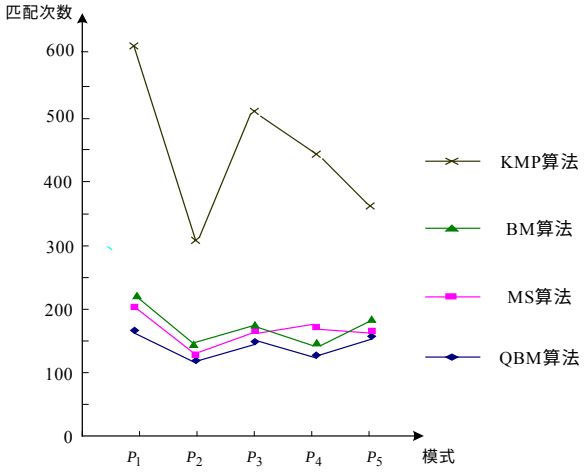


图 7 算法在不同模式串下的匹配次数

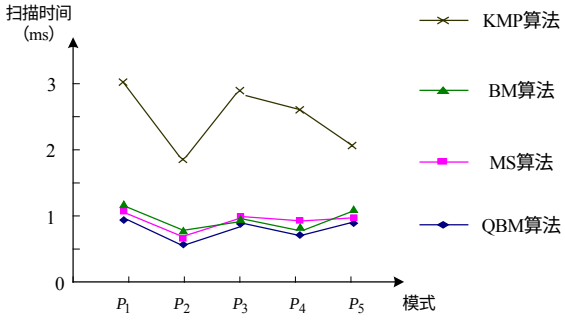


图 8 算法在不同模式串下的扫描时间

对实验结果分析如下：

(1)由图 6 可知，在同一个模式 P 下，QBM 算法在算法的运行时间和字符的匹配次数上都要小于其它 3 种算法，对测试结果进行处理得出，QBM 算法的匹配次数为 KMP 算法的 31%、BM 算法的 81%、MS 算法的 83%。

(2)由图 7 和图 8 可得，针对所有的模式 P，QBM 算法无论是在运行时间还是在所需的匹配次数上，都要明显优于其它算法。

3 结束语

本文设计的 QBM 算法采用了最长前缀和 shift 移动表的思想，其时间复杂度的理论值低于包含 BM 算法在内的其它常用串匹配算法，并且在实际的使用过程中充分展示了其相对高效的特点。

参考文献

1 Knuth D E, Morris J H, Pratt V R. Fast Pattern Matching in String[J]. SIAM Journal on Computing, 1977, 11(6): 323-350.

2 Boyer R S, Moore J S. A Fast String Searching Algorithm[J]. Communications of ACM, 1977, 20(10): 762-772.

3 Karp R M, Rabin M D. Efficient Randomized Patter Matching Algorithm[J]. IBM.J.Res.Develop, 1987, 31(3): 249-260.

4 Vishkin U. Deterministic Sampling——A New Technique for Fast Pattern Matching[J]. SIAM Journal on Computing, 1991, 20(11): 22-40.

5 朱　洪, 陈增武. 算法设计和分析[M]. 上海: 上海科学技术文献出版社, 1989: 135-137.

6 黄占友, 刘　悦. 对 KMP 串匹配算法的改进[C]. 第四次全国便携计算机学术交流会论文集. 北京: 科学出版社, 1997: 20-22.

7 贺龙涛, 方滨兴, 胡铭曾. 对 BM 串匹配算法的一个改进[J]. 计算机应用, 2003, 23(3): 6-8.

8 赵一谨. 一个改进的BM串匹配算法[J]. 计算机研究与发展, 1998, 35(1): 45-48.

（上接第 77 页）

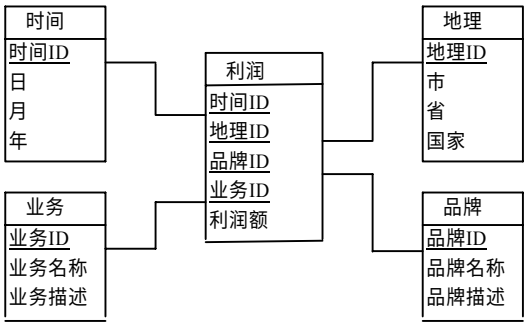


图 5 转换后的星型模型

4 总结

目前，数据仓库的设计方法主要有自底向上和自顶向下 2 种。在自底向上的设计方法中，面临一个重要问题就是如何在企业现有的底层数据库结构的基础上构建符合用户要求的多维模型。本文针对这种情况介绍了 HPKM，对其构建过程、维层次关系进行了探讨，并分析了 HPKM 到多维模型的转换方法。在数据仓库概念设计阶段，使用 HPKM 能更好地获取多维应用语义，从而为数据仓库概念模型设计提供一种新的思路。

参考文献

1 庄琴生. 以 E-R 模型为基础构造数据仓库的概念模型[J]. 计算机工程与应用, 2004, 40(10): 195-198.

2 唐九阳, 陆昌辉, 邓　苏等. ER 模型与多维模型互相转换的研究[J]. 计算机工程, 2003, 29(1): 97-99.

3 Golfarelli M. Conceptual Design of Data Warehouse from E/R Schemes[J]. Proceedings of the 32nd Hawaii International Conference on System Science, 1998.

4 Nectaria. StarER: A Conceptual Model for Data Warehouse Design[C]. Proceeding of DOLAP, 1999.

5 罗　军, 吕德文, 陈　松等. 基于 E-R 模型层次化的录入技术[J]. 重庆大学学报, 2003, 26(7): 21-23.