第 33 卷 第 5 期 计 算 机 工 程 2007 年 3 月
Vol.33 No.5 Computer Engineering March 2007
·软件技术与数据库· 文章编号：1000—3428(2007)05—0038—03 文献标识码：A 中图分类号：TP393.08

基于网络处理器的高效中英文多模式匹配算法

廖明涛，张德运，李金库
（西安交通大学电信学院网络研究所，西安 710049）

摘 要：由于中英文字符在编码方面的差异，传统面向英文字符环境的多模式匹配算法无法直接应用于中英文字符混合环境。提出了一种适用于网络处理器和中英文混合环境的高效多模式匹配算法。该算法采用从左向右的正向匹配，以字节为最小匹配单位，以字符为最小移位单位，在 Trie 树结构基础上，利用块字符匹配降低逐字匹配的概率，结合 Quick Search(QS)算法进行跳跃加速。实验表明，算法能够在中英文混合环境下避免字节错位和误匹配，匹配速度优于已有算法，且不存在空间膨胀问题，能够满足高速网络信息审计的要求。
关键词：网络处理器；多模式匹配；字符串匹配

High Efficiency Chinese-English Multi-pattern Match Algorithm Based on Network Processor

LIAO Mingtao, ZHANG Deyun, LI Jinku
(Institute of Network, School of Information, Xi'an Jiaotong University, Xi'an 710049)

【Abstract】 Due to different encoding of Chinese and English characters, traditional multi-pattern match algorithms do not work for Chinese-English characters mixed text. This paper proposes a high efficiency multi-pattern match algorithm for Chinese-English mixed environment based on network processor. The algorithm uses left to right positive direction match, considers one byte as least match unit and one character as least shift unit. Based on hashed Trie structure, it utilizes block characters match to reduce the probability of ordered character at a time match, uses Quick Search algorithm to shift and speed up the match. The experiment shows that the algorithm is faster than existing algorithms, avoids byte misplace and mismatch under Chinese-English mixed environment, satisfies the requirement of high performance network information audit system.
【Key words】 Network processor; Multi-pattern match; String match

1 概述

基于内容的网络信息审计系统通常部署在网络出入口等关键点，对网络传输数据进行实时分析、审计与控制，实现对网络内容的安全管理。传统基于通用处理器和 ASIC 的解决方案无法兼顾网络信息审计对高性能、灵活性与扩展能力的需求。作为一种专门为网络应用而优化设计的新型处理器，网络处理器综合了通用处理器和 ASIC 的优点，被认为是推动下一代网络向高性能和灵活性发展的关键技术。

多模式匹配是从文本串 T 中一次查找多个模式串 $p_1, p_2, \ldots, p_q$，其中 q 为模式串的数目，模式串 p_k 的长度为 m_k。多模式匹配是网络信息审计系统的关键技术，也是影响系统性能的瓶颈。

和英文字符不同，中文字符采用双字节编码。传统的多模式匹配算法如DFSA算法[1]、Wu-Manber算法[2]等都以字节为最小匹配和移位单位，只适合于单字节内码字符环境，在中英文混合环境下使用时，有可能将一个中文字分割为两部分，从而产生字节错位，并导致误匹配。组合状态DFSA算法[3]利用虚状态和实状态分别表示中文字符高字节和低字节的匹配状态，解决了中文字符构建完全哈希表时的空间膨胀问题，但该算法只适用于纯中文环境。两级哈希表DFSA算法[4]和线索化哈希Trie算法[5]较好地解决了中英文字符混合环境下的误匹配问题，但算法性能有待进一步改进。同时，已有的算法主要基于软件方式实现，在网络处理器环境下应用时，无法充分利用网络处理器的特点。

本文在文献[5]提出的中英文模式串 Trie 树结构基础上，通过结合块字符匹配和 Quick Search（QS）算法思想，提出一种基于网络处理器的适用于中英文混合环境的高效多模式匹配算法。

2 基于网络处理器的高效中英文多模式匹配算法

2.1 匹配方向与最小匹配移位单位

中文字符采用双字节编码，属于双字节字符集，第 1 个字节称为“高字节”，第 2 个字节称为“低字节”。中文编码的高字节的最高位为 1，而低字节的最高位则可能不为 1。在读取字符流时，如果遇到高位为 1 的字节，则将下两个字节作为一个双字节编码。因此，如果算法的匹配方向为从左向右的正向匹配，通过判断字节的最高位，即可区分中文字符和英文字符；反之，如果匹配方向为从右向左的反向匹配，则在当前匹配窗口内，根据当前字节以及其左侧的字节，有可能无法有效区分中英文字符。因此，匹配算法采用从左向右的正向匹配。在正向匹配过程中，算法以字节为最小匹配单位。对于 1 个中文字符，最大需要 2 次匹配；对于 1 个英文字符，需要 1 次匹配。

为了避免匹配指针移位所造成的字节错位，算法以字符为最小移位单位。即在移位过程中，如果当前字符为中文字符，匹配指针向右移动 2 个字节，并计为 1 个移位单位；如

基金项目：国家“863”计划基金资助项目(2003AA148010)；陕西省重大科技创新基金资助项目(2004ZKC01-09)
作者简介：廖明涛(1973－)，男，博士生，主研方向：计算机网络安全；张德运，教授、博导；李金库，博士、高工
收稿日期：2006-10-25 **E-mail：**kevinliao2001@gmail.com

果当前字符为英文字符，匹配指针向右移动 1 个字节，计为 1 个移位单位。

2.2 基于 Trie 树结构的多模式匹配

利用中英文模式串集合构建完全哈希Trie树结构的多模式匹配机，中文字符以其内码为键值，采用两级相邻的结点表示，英文字符用一级结点表示[5]。为加快查找速度，将所有Trie叶子结点表示为大小为 255 的完全哈希表。由于以字节为单位构造哈希表，匹配机不存在空间膨胀问题。Trie树中子结点的数据结构为：

```
typedef struct TrieNode{
  struct TrieNode *next[255];
  bool bOutput;
  int nPattern;
}TrieNode;
```

其中 next 表示结点的子结点，bOutput 表示当前结点是否有输出，如果有输出，nPattern 表示对应模式串在模式串集合中的索引。以模式串集合{中文、中国、中国人、IBM 公司}为例，构造的 Trie 树如图 1 所示。

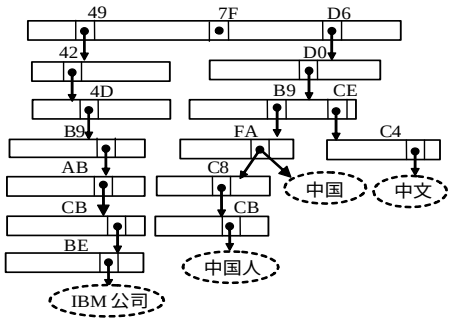


图 1 中英文模式串 Trie 树

基于 Trie 树的多模式匹配算法的基本思想是：在初始时，从文本串 T 的第 1 个字符开始，在 Trie 树上进行匹配查找；当本次匹配过程结束时，从文本串的第 2 个字符开始，在 Trie 树上进行匹配查找；依此类推，直至文本串结束。

基于 Trie 树的匹配算法虽然能够很好地解决中英文混合环境下的多模式匹配问题，但算法需对文本串中的每个字符进行顺序匹配，即在每次匹配失败或成功后，文本串匹配指针比上次匹配的初始位置前进一个字符，算法执行效率不高。

2.3 基于块字符匹配和 QS 算法的匹配跳跃

本节通过基于块字符匹配和QS算法思想，对基于Trie树的匹配算法进行改进，从而降低逐字匹配的概率，提高算法的执行效率。和单个字符相比，多个字符同时匹配失败的概率更大[2]。设模式串集合的最短长度为 *minlen* ，定义“块字符”长度为 *minlen* 的字符串所组成的“宽”字符。

考虑模式串的前 *minlen* 个字符，使用哈希函数将长度为 *minlen* 的块字符映射为一个整数，构造哈希表HASH。如果存在模式串前 *minlen* 个字符组成的子串的哈希值为h，则对应值HASH[h]=1，否则HASH[h]=0。本文哈希表选用适合于字符串的按位哈希函数[2,7]。

在匹配过程中，设当前文本串的匹配指针位置为 *i*，首先计算字符串 $t_i \cdots t_{i+minlen-1}$ 的哈希值 h，如果 HASH[h]值为 0，则匹配失败，本次匹配过程结束，指针后移，进入下一个匹配循环；如果 HASH[h]为 1，则利用 Trie 树进行匹配。

在网络处理器环境下，利用“块字符”匹配主要有两方面优点：

(1)由于块字符匹配失败的概率更大，因此进行块字符匹配，能够显著降低利用 Trie 树进行逐字顺序匹配的概率，从而提高整体运行速度；

(2)网络处理器通常采用分级内存管理机制，如在 Intel IXP2400 中，有本地存储器、SRAM、DRAM 等 3 种内存资源。SRAM 和 DRAM 为片外存储，具有较大的容量，但速度较慢，存取周期一般为 90 和 120 个时钟周期；本地存储器为微处理器片内空间，速度非常快，其存取时间最多仅为 3 个时钟周期，但容量较小。由于 Trie 树结构需要存储大量的模式串，通常只能放置到片外 SRAM 中。通过块字符匹配方法，构造容量较小的哈希表 HASH，并将其存储到片内存储器中。在匹配时，首先读取片内 HASH 表，只有当 HASH 表命中时，才读取片外 SRAM 中的 Trie 树进行顺序匹配，从而提高匹配速度。由于微处理器的片内存储器空间有限，将 HASH 表中的每一项用一个比特位存储，一个大小为 8192 的 HASH 表将占用 1 024B 的存储空间。

在基于Trie树的匹配中，每次匹配循环结束后，匹配指针向右的移动距离至少为 1，即当前块字符的下一个字符 $t_{i+minlen}$必定包含在下一次匹配过程中。基于QS算法思想[6]，利用该字符带来的信息，可以加大跳跃距离。

考虑所有模式串的前 *minlen* 个字符，即假设所有模式串的长度为 *minlen* ，形成模式串集合P_{minlen}。根据字符$t_{i+minlen}$在集合中的出现信息，跳跃距离的计算可分为两种情况：

(1)如果字符$t_{i+minlen}$在模式串集合P_{minlen}不出现，则最大安全跳跃距离为 $minlen+1$。即此时匹配指针跳跃到字符$t_{i+minlen}$的下一个位置；

(2)如果字符$t_{i+minlen}$在模式串集合中出现，则最大安全跳跃距离为字符$t_{i+minlen}$在P_{minlen}中的最右出现距离。即此时匹配指针向右跳跃，使得字符$t_{i+minlen}$和其在模式串集合P_{minlen}中的最右出现位置 对齐。

跳跃距离 $skip(c)$的计算如下：

$$skip(c)=\begin{cases} minlen+1, & P_{minlen}\ not\ contain\ char\ c \\ \min\{minlen-j \mid p_k[j]=c, 0\le j<minlen, 1\le k\le q\}, & \\ P_{minlen}\ contain\ char\ c & \end{cases} \quad (1)$$

在网络信息系统中，模式串集合通常都预先确定，因此跳跃距离 $skip(c)$的值可以预先计算，并形成跳跃表 SKIP。

2.4 基于网络处理器的实现

算法中主要有 3 种数据结构：Tire 树，HASH 表和 SKIP 表。Trie 树中需要存储模式串集合，将其存放在 SRAM 中，为加快首字节匹配的效率，将 Trie 树第 1 层的 255B 数据存储在本地存储器中；SKIP 表的大小为 64KB 字节，存储在 SRAM 中；HASH 表的大小为 1 024B，将其作为程序代码的一部分，存储到微处理器内部的指令空间中。

算法主要包括 3 个执行部分：基于 SKIP 表的跳跃距离计算，基于 HASH 表的块字符匹配和基于 Trie 树的模式串搜索。利用网络处理器的硬件多线程技术，可以将执行部分分配到不同的线程中，降低内存访问延迟对系统性能的影响。

从输入端口接收的数据包数据，存储到外部 DRAM 中。在多模式匹配时，需要对数据包数据进行大量的操作，为提高算法执行速度，引入线程将存储在 DRAM 中的数据包数据预取至本地存储器，对数据包的后续处理都在本地存储器中执行。

2.5 算法性能分析

Trie 树构造算法性能：构造 Trie 树需要遍历模式串中的每个字节，算法的时间复杂度为$O\left(\sum_{k=1}^{q} m_k\right)$。在最坏的情况下，所有的模式串都不存在公共前缀，Trie 树所占用的空间大小为$\Omega = 255 + \sum_{k=1}^{q}(m_k - 1)*255$，能够满足网络处理器环境下的空

间复杂度需求。

实时处理性能：在最优情况下，匹配算法的时间复杂度为 $O\left(\frac{n}{minlen+1}\right)$，在最差情况下，算法的时间复杂度为 $O(n*maxlen)$。在平均情况下，算法的时间复杂度与字符出现的概率有关。首先分析QS跳跃对算法性能的影响。设文本串和模式串中的所有字符均来自于大小为 δ 的字符表 Ω，模式串集合 P_{minlen} 所包含的不重复的字符个数为 δ'。为简化计算，假设字符表 Ω 中所有字符具有相等的出现概率，则对于文本串中的任意字符 c，其在 P_{minlen} 中出现的概率为 $pb_t = \delta'/\delta$。设 c 在 P_{minlen} 中各个位置出现的概率相等，则平均跳跃距离为

$$sk = (1-pb_t)*(minlen+1) + pb_t * \frac{(minlen+1)}{2} \quad (2)$$
$$= \left(1-\frac{pb_t}{2}\right)*(minlen+1)$$

中文为大字符集语言，字符 δ 值比较大，如果模式串数目适中，pb_t 值通常较小，平均跳跃距离将接近于 $minlen+1$，即在平均情况下，算法的时间复杂度接近于最优情况。

对于块字符匹配，由于在中文环境下，长度为 *minlen* 的“块字符”在 HASH 表中命中的概率非常低，因此在绝大部分情况下，在“块字符”匹配后，本次匹配循环结束，不需要根据 Trie 树进行逐字顺序匹配。

3 实验

利用Radisys公司基于Intel IXP-2400 网络处理器的接口卡ENP-2611[8]对本文算法进行实验。实验采用 500 个模式串组成的集合，包括 400 个中文模式串和 100 个英文模式串。中文模式串的平均长度约为 4 个中文字符，英文模式串的平均长度约为 6。在IXP2400 中，采用 4 个微引擎完成本文算法：微引擎 1 负责数据包接收，微引擎 2 负责数据包分类，微引擎 3 和 4 从处理队列获取待处理数据包，进行多模式匹配。实验配置微引擎 2 将所有的数据包都添加到处理队列中进行处理，匹配完成后，丢弃数据包，并统计完成匹配的数据包数量。

我们对本文算法和基于哈希 Trie 的多模式匹配算法（未线索化）进行了实验。实验采用 SmartBits6000 网络测试设备以千兆速率向输入端口发送待审计的数据包，发送持续时间为 120s，分别测试了 64B、128B、256B、512B、1 024B、1 518B 时的系统性能，结果如表 1 所示。从表中可以看出，采用了块字符匹配和 QS 算法加速后，处理性能得到较大幅度提高。

表 1 算法实验结果

包长（B）	发送速度（包/s）	本文算法处理包数(%)	Trie 树匹配算法处理包数（%）
64	1 488 095	43.67	21.05
128	84 4595	45.48	23.06
256	452 899	48.92	24.63
512	234 962	51.13	25.32
1 024	119 732	53.04	26.94
1 518	81 274	53.88	27.21

4 结论

本文提出了一种基于网络处理器的适用于中英文混合环境的高效多模式匹配算法。该算法采用从左向右的正向匹配，以字节为最小匹配单位，以字符为最小移位单位，在 Trie 树结构基础上，利用块字符匹配降低逐字匹配的概率，同时结合 QS 算法进行跳跃加速。理论分析和实验表明，本文算法能够在中英文混合环境下避免字节错位和误匹配，匹配速度明显优于已有的中英文多模式匹配算法，且不存在空间膨胀问题，能够满足高速网络信息审计的要求。

参考文献

1 Aho A V, Corasick M J. Efficient String Matching: An Aid to Bibliographic Search[J]. Communications of the ACM, 1975, 18(6): 333-340.

2 Wu S, Manber U. A Fast Algorithm for Multi-pattern Searching[R]. University of Arizona, TR: 94-17, 1994.

3 高　鹏. 基于内容的网络信息审计及其关键技术研究[D]. 西安: 西安交通大学, 2003.

4 沈　洲, 王永成, 许一震. 一种面向中文的快速字串多模式匹配算法[J]. 上海交通大学学报, 2001, 35(9): 1285-1289.

5 孙钦东. 网络信息内容实时审计及其关键技术研究[D]. 西安: 西安交通大学, 2005.

6 Sunday D M. A very Fast Substring Search Algorithm[J]. Communications of the ACM, 1990, 33(8): 132-142.

7 Mckenzie B J, Harries R, Bell T. Selecting a Hashing Algorithm[J]. Software Practice and Experience, 1990, 20(2): 209-224.

8 Corporation R. ENP-2611 Data Sheet[EB/OL]. 2004. http://www.rad isys. com/files/data_sheets/ENP-2611_07-1236-06_ 0506_datasheet.pdf.

（上接第 37 页）

3 结束语

通过该模型的分析，可以较好地解决基于交通网络的移动对象管理的相关问题，伴随 GPS 精确率的提高，该模型对于移动对象的管理可以进一步精确，还用以解决交通意外分析、交通堵塞的提前预防、车辆闯红灯等违规操作的自动记录等。其中存储的移动对象历史移动数据是对移动对象的行为模式等进行数据挖掘的宝贵资源。本文的创新点在于：对于基于公路网的移动对象的运动模式进行了深入研究，采取了一种实时平均法反映每条交通路段的交通情况，能够对移动对象进行有效的管理和预测。

参考文献

1 孟小峰, 周龙骧, 王　珊. 数据库技术发展趋势[J]. 软件学报, 2004, 15(12): 1822-1836.

2 Sistla P, Wolfson O. Modeling and Querying Moving Objects[C]// Proceedings of the 13th International Conference on Data Engineering. 1997.

3 Song M. Mobility Patterns in Moving Objects Database[C]//Proc.of the International Conference on Parallel Processing. 2003.

4 Chen Ying, Rao Fangyan. Managing Location Stream Using Moving Object Database[C]//Proceedings of the 14th International Workshop on Database and Expert System Applications. 2003.

5 Beiyin H, Wolfson O. A Weight-based Map Matching Method in Moving Objects Databases[C]//Proceedings of the 16th International Conference on Scientific and Statistical Database Management. 2004.

6 张健奇, 蔡菲娜. 集中差分技术在城市车辆智能交通管理中的应用[J]. 微计算机信息, 2005, 21(3): 190-191.