基于文本片断的多模部分匹配算法

廖唯荣，邹维
(北京大学 计算机科学技术研究所，北京 100871)

摘 要：在进行基于关键词的网络内容检测时，通常需要流重组甚至文件重组，这将造成安全网关检测时延过大。针对该问题，首先提出了基于文本片断的部分匹配概念，然后设计并实现了一种基于文本片断的多模部分匹配算法。该算法能检测文本片断中的特征片断而无需进行重组。最后经过实验验证，相对于基于完整文本的多模匹配算法，在文本片断足够大的情况下，该算法漏报率为 0，且误报率足够小。

关键词：信息安全；多模匹配；部分匹配；网络内容检查
中图分类号：TP301 **文献标识码**：A **文章编号**：1000-436X(2010)03-0031-06

Multi-keyword partial matching algorithm based on text fragments

LIAO Wei-qi, ZOU Wei
(Institute of Computer Science & Technology of Peking University, Beijing 100871, China)

Abstract: Network content detection based on keywords in security gateway often need to reassemble the TCP packets and even the whole file, which will cause a long delay. In order to resolve the problem, a new concept of pattern partial-matching based on text fragments was presented. A novel multi-keyword partial matching algorithm was also designed and realized which could detect the fraction of keywords in data block without re-flow or re-file. Comparing to the normal multi-keyword matching algorithm, it was proved by experiments that the false negative of the algorithm is zero, and the false positive rate is little enough when the data block is large enough.

Key words: information security; multi-keyword matching; partial matching; Internet content detecting

1 引言

模式匹配问题研究在信息检索、模式识别等众多领域均有重大价值。尤其在信息安全的诸多应用中，如入侵检测、内容过滤、计算机病毒特征码检测等。根据其一次扫描中匹配的模式数，模式匹配可分为单模匹配和多模匹配 2 种。单模匹配有 KMP 算法和平均性能优化的 Boyer-Moore 算法[1]等。与单模匹配有所不同，多模匹配算法可以在一次扫描中匹配大量的模式——通常有上万个模式[2]，因而得到了更加广泛的应用。

常用的多模匹配算法都要求被检测文件的完整性，因此在网络环境下，为了避免因待检测特征模式被分割在不同的 IP 数据分组中而造成漏报，通常需要进行网络层流重组甚至应用层文件重组。这不仅造成网络设备进行检测时的时延过大，而且这种重组方法也受到技术上的挑战。1）应用层文件重组依赖于具体的应用协议，如 HTTP 协议的文件重组方法不能应用到 FTP 协议的文件传输过程。网络设备需要对不同的文件传输协议进行不同的处

收稿日期：2009-09-02；修回日期：2009-12-05
基金项目：国家高技术研究发展计划（“863”计划）基金资助项目（2006AA01Z410）；国家 242 信息安全研究专项（2006A13）
Foundation Items: The National High Technology Research and Development Program of China (863 Program) (2006AA01Z410); The Special Project on Research of National Information Security (2006A13)

理，这不但代价大，而且缺乏灵活性。2）网络层流重组虽然避免了对应用层协议的依赖，但 P2P 等文件传输协议可能将一个文件的不同分块利用不同的流进行传输，流重组本质上只是将小片断变成大片断，因而仍然没有达到被检测文件的完整性要求。

为了减小网络设备内容检测的延迟，无需流重组或文件重组的流式扫描方法备受关注。目前，尽管个别网络安全厂商声称能做到流式内容检测，但尚未见到有说服力的文献。本文另辟蹊径，提出部分匹配的概念，设计并实现了在待查文件的片断中进行多模式部分匹配的算法。与基于完整文件的多模匹配算法相比，基于部分匹配的方法不会发生漏报，而且在匹配关键词不太短的情况下，其误报率也能得到较好的控制。

2 相关工作

Aho 和 Corasick 于 1975 年最早提出 AC 算法[3]。它根据多个模式串构建一个 Trie 树，然后在 Trie 树上利用广度优先方式得到失效转移函数，从而得到完整的多模匹配自动机。后来围绕这一算法，一些学者提出了改进算法：Commentz-Walter[4]将 AC 算法和 BM 算法结合起来，利用跳跃扫描技术提高匹配效率。Tuck 和 Sherwood[5]等实现了一个基于位图的算法，有效的压缩了存储空间。Wang 等[6]利用反向构建自动机和跳跃扫描启发匹配等技术提出了一个优化方法。Wu 和 Manber[2]于 1994 年提出了基于后缀匹配的方法，采用了 Hash 表和跳跃扫描技术来加速匹配，因此 WM 算法也成为了当前众多主流软件所采用的多模匹配算法。Li 等[7]在分析以上几个算法的基础上，介绍了利用 FPGA、Bloomfilter 或 TMAC 等硬件实现多模匹配的方法和策略。此外，文献[8~12]介绍了在中文或中英文混编环境下的多模匹配算法。

近来，多模相似匹配成为多模匹配研究中一个颇受关注的问题。Muth 和 Manber[13]利用两级散列提出了允许一个错误的多模相似匹配算法。Ricardo 和 Gonzalo 在文献[14~16]中对此算法进行了扩展，使其可以容忍多个错误。在文献[17]中，他们分别基于并行比特串搜索状态向量、文本窗中的模式字符计数、对已有多模式精确匹配的扩展(分解模式串)，实现了 3 种针对英文字符串的相似匹配算法。Gao 等[18]基于 Episode 距离和多个模式的有限状态机，实现了允许若干插入错误的多模相似匹配算法。

本文即将描述的算法是多模匹配算法在特定需求下的一种应用。其做法类似于相似匹配，但又有很大的区别。其相同点在于：1）两者都应用于待检测文本完整性受到破坏的条件下；2）两者都不需要对模式进行精确匹配，即允许匹配有一定范围的偏差。但两者也有所区别：1）前者待检测文本完整性破坏缘于对文本的分割，而后者缘于对其内容有意识的篡改；2）前者进行多模匹配发生漏报，总是在文本片断的首尾，而后者可能发生于待测文本的任何位置。3）前者对模式片断的匹配仍然要求精确匹配（即部分匹配），而后者可以是对模式多处增删改后的匹配（即相似匹配）。

3 基于文本片断的部分匹配及其性质

为了方便讨论，对多模匹配形式化描述为：给定多个模式的集合$\{P_1, P_2, \cdots, P_k\}(k\geqslant 2)$，这里 k 称为模式个数，模式 P_i 用字节串 $b_1b_2\cdots b_m$ 表示，其中 m 称为模式 P 的长度；用字节串 $a_1a_2\cdots a_n$ 表示待匹配的文本 T，其中 n 称为文本 T 的长度。若文本 T 中的某字串 $a_ja_{j+1}\cdots a_{j+m-1}$ 与模式 P_i 的各个字节一一对应，则称 T 从 j 开始匹配 P_i，或称 P_i 匹配 T 于 j。

定义 1 若文本 T 中的某字串 $a_ja_{j+1}\cdots a_{j+r-1}$ 与模式 P_i 的前 r 个字节 $b_1b_2\cdots b_r$ 一一对应，则称 T 从 j 开始前缀匹配 P_i，匹配长度为 r；若文本 T 中的某字串 $a_ja_{j+1}\cdots a_{j+r-1}$ 与模式 P_i 的后 r 个字节 $b_{m-r+1}b_{m-r+2}\cdots b_m$ 一一对应，则称 T 从 j 开始后缀匹配 P_i，匹配长度为 r。若 T 从 1 起后缀匹配 P_i，则称 T 后缀首匹配 P_i；若 T 从 $n-r+1$ 起前缀匹配 P_i，则称 T 前缀末匹配 P_i。这 2 种特征匹配都称为 T 部分匹配 P_i，并将一般的 T 匹配 P 称为完全匹配。

如图 1 所示，文本被分成了 3 块，左边一块前缀末匹配特征串①，右边一块后缀首匹配特征串③，中间这一块不但完全匹配特征串②，还部分匹配特征串①③。由此可以得到如下定理。

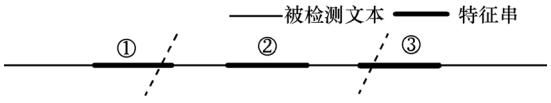

图 1 基于文本片断的模式匹配图

定理 1 在多模匹配中，模式集合$\{P_1, P_2, \cdots, P_k\}$中最大模式长度为 m，文本 T 可能会被分成若干部分$\{T_1, T_2, \cdots, T_s\}$，并约定每一部分 T_j 都不小

于最大模式长度 m；那么 T 与某模式 P_i 匹配的必要条件是$\{T_1, T_2, \cdots, T_s\}$中存在某个数据块 T_j 满足下列 3 个条件之一：

① T_j 完全匹配模式 P_i，其匹配长度为 m_i；

② T_j 前缀末匹配模式 P_i，其匹配长度不小于 $m_i/2$；

③ T_j 后缀首匹配模式 P_i，其匹配长度不小于 $m_i/2$。

证明　设 $T=a_1a_2\cdots a_n$，$P_i=b_1b_2\cdots b_m$，当 T 与 P_i 匹配时；根据定义 T 中子串 $a_ja_{j+1}\cdots a_{j+m-1}$ 与模式 P_i 的各个字节 $b_1b_2\cdots b_m$ 一一对应。

因为在 T 的分块$\{T_1, T_2, \cdots, T_s\}$中，每一部分 T_j 都不小于模式长度 m，所以子串 $a_ja_{j+1}\cdots a_{j+m-1}$ 在$\{T_j\}$中的分布有 2 种可能，其中一种是整个子串落到同一个分块 T_j 中，另一种是该字串落到 2 个相继的分块中，不妨设为 T_j 和 T_{j+1}。下面分别讨论这 2 种情况。

1) 当子串 $a_ja_{j+1}\cdots a_{j+m-1}$ 落在一个分块 T_j 中时，T_j 完全匹配模式 P_i，即条件①满足；显而易见条件①是 T 匹配 P_i 的充分条件。

2) 当子串 $a_ja_{j+1}\cdots a_{j+m-1}$ 落在相继的 2 个分块 T_j 和 T_{j+1} 中时，其中 T_j 部分的长度为 r，T_{j+1} 部分的长度为 $m–r$，则有下面 2 种情况：

i) 当 $r\geqslant m_i/2$ 时，即 T_j 中有子串 $a_ja_{j+1}\cdots a_{j+r-1}$，该字串与 $b_1b_2\cdots b_r$ 一一对应，所以 T_j 前缀末匹配模式 P_i，且匹配长度 r 不小于 $m_i/2$，即条件②满足；

ii) 当 $r<m_i/2$ 时，则 T_{j+1} 中有子串 $a_{j+r}a_{j+r+1}\cdots a_{j+m-1}$，该子串与 $b_{r+1}b_{r+2}\cdots b_m$ 一一对应，所以 T_{j+1} 后缀首匹配模式 P_i，且匹配长度 $m–r$ 不小于 $m_i/2$，即条件③满足。

综上所述，定理得证。　□

从定理 1 可以看到，如果多模匹配算法能在文本 T 中匹配出模式 P_i，则一定可以在 T 的一个分块 T_j 中匹配模式 P_i 的一个长度不小于 1/2 的子模式长度，而且如果匹配上的是 P_i 的一个真子模式，那么匹配子串必定位于 T_j 的首部或尾部。

应该注意到，前缀末匹配中所用的特征串前缀，在多模匹配有限状态机中与特征串的状态是重合的；但是后缀首匹配中所用的特征串后缀，不是从特征串的起始位置开始的，因此需要把不同长度的后缀作为一个特征加入状态机。这种做法会导致状态机的状态数从 O(km)量级上升到 O(km^2)量级。为了避免这种状态数目剧增，可以用 P^{-1} 代替 P 的所有后缀加入状态机，但是在扫描过程中，需要对文本块的前 m 个字节做一个逆向扫描。用 T^{-1}，P^{-1} 表示文本和模式的逆。例如：T=abcdef，则 T^{-1}=fedcbca。那么可以得到下面一个定理。

定理 2　T 前缀末匹配 P，等价于 T^{-1} 后缀首匹配 P^{-1}。

证明　设 $T=a_1a_2\cdots a_n$，$P=b_1b_2\cdots b_m$，则 $T^{-1}=a_na_{n-1}\cdots a_1$，$P^{-1}=b_mb_{m-1}\cdots b_1$。

先证充分条件(→)，若 T 前缀末匹配 P；根据定义 T 中有长 r 的子串 $a_1a_2\cdots a_r$ 与模式 P 的子串 $b_{m-r+1}b_{m-r+2}\cdots b_m$ 一一对应。

即 2 个子串的逆 $a_ra_{r-1}\cdots a_1$ 与 $b_mb_{m-1}\cdots b_{m-r+1}$ 也一一对应。

因为上面 2 个串分别是 T^{-1} 的末子串和 P^{-1} 的起始字串，所以根据定义，T^{-1} 后缀首匹配 P^{-1}。

同理可证必要条件（←）。证毕。　□

下一节将根据这 2 个定理，设计基于部分匹配概念的流式多模匹配算法。

4　流式多模匹配方法的结构和算法设计

本文将仿照 AC 算法，构造 Trie 树并实现模式匹配自动机，得到转移函数 goto 和失效转移函数 fail；重新定义匹配结果函数 output 使其适于部分匹配结果的输出；并设计多模部分匹配算法，称之为 Partial-AC 算法。为了给出直观的表示，下文将采用模式集合{*he*，*she*，*his*，*hers*}作为例子，该模式集的最大特征串长度为 4。

4.1　确定型有限状态自动机的构造

转移函数 goto 和失效函数 fail 都是为了在匹配状态机中指示状态转移用的，其意义与 AC 算法中一致。与 AC 算法不同的是，在本文构造多模式有限状态机时，为了找到文本分块与模式的后缀首匹配，不但要加入原有的每个模式 P_i，还要加入 P_i 的逆 P_i^{-1}。但是为了避免重复匹配和减少状态数，P_i^{-1} 的最后一个字符不需要加入状态机。如图 2 所示，Trie 树由模式集{*he*，*she*，*his*，*hers*}和模式逆的集{*e*(*h*)，*eh*(*s*)，*si*(*h*)，*sre*(*h*)}的元素符号组成。

4.2　Output 函数的设计

Output 函数的值域为一个向量集合，集合的元素是一个匹配结果，其内容包括：匹配的模式，模式匹配的方向，模式匹配的长度，记为向量<*pat*，*dir*，*len*>，简记为 $pat^{+/-\text{len}}$。由此可以得到例子模式集{*he*，*she*，*his*，*hers*}的 output 函数值。

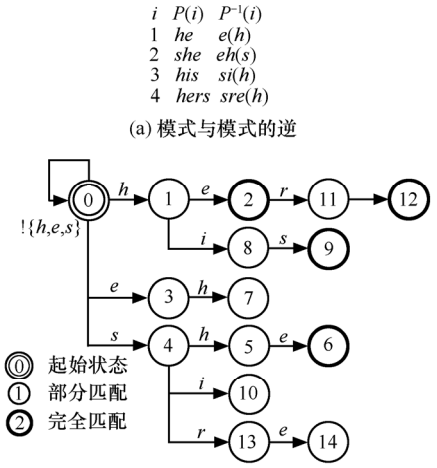

图 2　Partial-AC 算法自动机示意图

本文算法的 output 函数中，14 个状态共计 40 个部分匹配结果。但是因为前缀末匹配和后缀首匹配是相伴存在的，即在某一文本块中有前缀末匹配，则在相继文本块中有后缀首匹配。所以根据定理 2，只要考虑正向匹配长度不小于 1/2 模式长度，或反向匹配长度大于 1/2 模式长度的那个部分匹配。经过这一优化，表中的匹配结果减少为 16 个，如表 1 所示。

表 1　模式集{*he*，*she*，*his*，*hers*}的 output 函数值

i	output(*i*)	*i*	output(*i*)
1	$\{he^{+1}\}$	8	$\{his^{+2}\}$
2	$\{he^{+2},hers^{+2}\}$	9	$\{his^{+3}\}$
3	Φ	10	$\{his^{-2}\}$
4	Φ	11	$\{hers^{+3}\}$
5	$\{she^{+2},he^{+1}\}$	12	$\{hers^{+4}\}$
6	$\{she^{+3},he^{+2},hers^{+2}\}$	13	Φ
7	$\{she^{-2},he^{+1}\}$	14	$\{hers^{-3}\}$

4.3　编译算法设计

编译算法是用来构造 goto 函数、fail 函数和 output 函数，其目的是根据模式集形成一个多模式有限状态自动机(multi-pattern FSM)。下面的算法 1 用于构造多模式匹配的 goto 函数和部分 output 函数。

算法 1　构造 goto 函数

输入：模式集$\{P_1, P_2, \cdots, P_k\}(k\geqslant 2)$。

输出：用于多模式匹配的 goto 函数和部分 output 函数。

方法：

newstate ← 0；

for i ← 1 to k do

　enter (P_i，“+”)；　//将模式 P_i 加入多模自动机

　enter (P_i，“−”)；　//将模式 P_i^{-1} 的后 $m-1$ 字节加入多模自动机

for all a st.goto (0，a) = fail do goto (0，a) ← 0；

procedure　enter ($b_1b_2\cdots b_m$，dir)

　state ←0；if dir = “+” then j=1；eles j=m；

　while goto ($state$，b_j)<>fail do //遍历重叠节点

　　state ← goto ($state$，b_j)；

　　if dir = “+” then

　　　output ($state$) = output ($state$) ∪

　　　　{<P_i，“+”，j>}；　j ← j+1；

　　else

　　　output ($state$) = output ($state$) ∪

　　　　{<P_i，“−”，$m-j+1$>}；　j ← $j-1$；

　while (dir= “+” and j<=m)or(dir= “−” and j>1)

　do　//建立新节点

　　newstate ← newstate +1；

　　goto ($state$，b_j)←newstate；state←newstate；

　　if dir = “+” then

　　　output($state$) = output ($state$) ∪

　　　　{<P_i，“+”，j>}；j ← j+1；

　　else

　　　output ($state$) = output ($state$) ∪

　　　　{<P_i，“−”，$m-j+1$>}；j ← $j-1$；□

fail 函数和 output 函数的另一部分的构造是利用广度优先的方法得到。这与 AC 算法是一致的，因此本文不再复述。

4.4　匹配算法设计

算法 2　无重组多模匹配算法

输入：待检测的文本分块 $T_s=a_1a_2\cdots a_n$，由模式集$\{P_i\}$编译好的多模匹配自动机 M=(goto，fail，output)，其中 goto(i, a)是正常转移函数，fail(i)是错误转移函数，output(i)是匹配结果函数。

输出：匹配上的模式 P_i，匹配长度 r，在 T_s 上的起始匹配位置 index。

方法：

state ← 0；

for i ← 1 to n do

　while goto($state$，a_i) = failed do

```
        state ← fail(state);
    state ← goto(state，ai);
    //输出完全匹配结果
    show ( output(state)，“+”，“=”，i );
  end for
  //输出前缀末匹配结果
  show (output(state)，“+”，“<”，n );
  state ← 0;
  for i ← m downto 1 do
    while goto(state，ai) = failed do
      state ← fail(state);
    state ← goto(state，ai);
  end for
  //输出后缀首匹配结果
  show (output(state)，“−”，“<”，1);
  procedure show (rstset，dir，type，idx)
    if rstset = empty then return;
    for each ele in rstset do
      if dir<> ele.dir then continue;
      if type=“=” and ele.len=len(ele.pat) then
        //完全匹配
        print ele.pat，ele.len，idx−ele.len+1;
      else if len(ele.pat)<= ele.len<len(ele.pat)
          and type=“<” then    //部分匹配
        if dir=“+” then pos=idx−ele.len+1
        else pos=idx;
        print ele.pat，ele.len，pos;                □
```

在上面的算法描述中，show 方法用于显示 state 状态所对应的 output 中记录的成功模式匹配的结果。函数的第 2 个参数标识匹配方向，“+”表示只显示正向匹配结果，“−”表示只显示逆向匹配的结果；函数的第 3 个参数标识完全匹配或部分匹配，“=”表示只显示完全匹配的结果，“<”表示只显示部分匹配的结果。最后一个参数用于计算匹配的起始位置。

5　流式多模匹配算法性能分析

本节将对编译出的状态机的空间复杂度、匹配算法时间复杂度、漏报率和误报率进行分析。为了便于分析，明确参数如下：模式集中共包含 k 个模式串，模式串的平均长度为 m_{ave}；待检测文本总长 n，检测时文本被分割成 d 个分块检查，平均每块长度为 $n_{blk}=n/d$。

5.1　编译过程性能分析

对于模式编译过程，本算法与 AC 算法相比，增加了对模式逆的处理，因此模式匹配自动机的状态数约为 AC 算法的 2 倍。output 函数中结果集的规模增加较多，因为能匹配长 $m_{ave}/2$ 特征字串的状态都有成功部分匹配的结果，且它可同时匹配模式和模式的逆，所以其规模约为 AC 结果集规模的 m_{ave} 倍。本文以随机的 80 个 32 字节的字节串作为模式，分别用 AC 算法和 Partial-AC 算法进行编译，其状态数和结果集数如表 2 所示。

表 2　80×32 字节模式集编译性能比较

算法	状态数	完整匹配结果数	前缀末匹配结果数	后缀首匹配结果数	总计
AC	2 546	80	—	—	80
Partial-AC	4 999	80	1 302	1 218	2 600

5.2　匹配过程性能分析

文本扫描过程中，需要增加模式长度的逆向扫描，所以扫描总长度为 $dm_{ave}+n$，是 AC 算法扫描长度 n 的 $1+m_{ave}/n_{blk}$ 倍。考虑到算法 2 为了找出完全匹配结果，需要在状态机的每一个状态遍历结果集，这无疑影响算法效率。只要在 output 函数中加入一个标志，标记该结果集是否包含完全匹配结果，即可提高算法性能。

采用 Dell Optiplex 330 的 PC 机进行实验比较，其配置 CPU 为 Intel Pentium Dual E2160，主频 1.8GHz，内存为 1.98GB。待检测文本是 135MB 的网络数据流样本，模式集是从样本中随机提取的 80 个长度为 32 字节关键词，理论匹配模式 46 304 个。实验结果如表 3 所示。

表 3　匹配算法性能比较

算法	匹配模式数	访问状态数	访问结果个数	匹配耗时
AC	40 657	135 877 081	41 578	8.64s
Partial-AC	46 309	153 392 491	47 414	10.4s

从表 3 的实验结果可以看到，Partial-AC 算法比预期的要慢，那是因为其数据结构比 AC 复杂，需要更多的 IO 操作造成的。尽管如此，Partial-AC 扫描过程效率仍和 AC 算法属于同一个复杂量级，符合网络内容检查过程中多模匹配的需要。

5.3　漏报率和误报率分析

检测过程中的漏报主要受文本分块长度和模式长度相互关系的影响。根据定理 1 可知，当文本

分块长度大于模式长度，即 $n_{blk} \geq m_{ave}$ 时，无论如何都不可能有匹配长度不小于模式 1/2 的部分匹配，所以算法漏报率为 0；当文本分块长度小于模式长度的 1/2，即 $n_{blk} \leq m_{ave}/2$ 时，不可能匹配到长度超过模式 1/2 的特征片段，所以漏报率为 100%；当文本分块长度在两者之间，即 $m_{ave}/2 \leq n_{blk} \leq m_{ave}$ 时，如果特征串落在 3 个文本块中，就会发生漏报，所以漏报率应为 $m_{ave}/n_{blk}-1$。

在本文的算法中，完全匹配是不会导致误报的，误报必然发生在部分匹配的情况下。那么很容易得到误报率的上限是部分匹配发生的概率，即 $2m_{ave}/n_{blk}$。

为了研究模式长度和文本分块长度对误报率的影响,取一个 135MB 的文件作为待测文本，并取出长度分别为 2，4，8，16，32 字节的 5 组特征串，每组特征 80 个。分别用 AC 多模匹配和本文的流式多模匹配对文件不同分块大小进行检测，得到的模式匹配数，如表 4 和表 5 所示。

表 4 模式长度对漏报率、误报率的影响(n_{blk}=256B)

模式长度/B	理论匹配数	AC匹配数	Partial-AC匹配数	AC漏报率	Partial-AC误报率
2	303 395	302 235	516 753	0.38%	70.32%
4	46 362	45 798	47 370	1.22%	2.17%
8	46 349	45 062	46 349	2.78%	0%
16	46 329	43 553	46 329	5.99%	0%
32	46 304	40 657	46 309	12.19%	0.011%

表 5 文本分块长度对漏报率、误报率的影响(m_{ave}=32B)

分块长度/B	理论匹配数	AC匹配数	Partial-AC匹配数	AC漏报率	Partial-AC误报率
32	46 304	1 461	46 335	96.84%	0.067%
64	46 304	23 880	46 318	48.43%	0.030%
128	46 304	35 061	46 313	24.28%	0.019%
256	46 304	40 657	46 309	12.19%	0.011%
1 024	46 304	44 888	46 306	3.06%	0.004%
1 600	46 304	45 387	46 306	1.98%	0.004%

从表 4 中可以看出：虽然较短的模式被分割到 2 个文本块中的概率比较低，但因为其表征力不强，容易造成误报，所以误报率比较高。从表 5 中可以看出，当文本分块比较小的时候，容易造成大量的部分匹配，AC 算法漏报巨大，而 Partial-AC 算法的误报率能控制得比较低。

6 结束语

为了实现网络内容检测中的流式扫描，本文一方面提出了基于文本片断的部分匹配的概念，设计了多模部分匹配算法，并从理论上证明了该算法的有效性；另一方面，本文实现了该算法，并通过实验研究了特征串长度和文本分块大小对匹配效果的影响，分析了该算法可能造成的漏报和误报。将该方法用于 P2P 网络的内容检测，取得了很好的检测效果。但同时也应该注意到，当病毒等恶意代码采用碎片攻击，即将其主体内容拆成小于模式长度的碎片进行发送的时候，采用部分匹配的方式仍无法探知，这是以后必须要克服的困难。

参考文献:

[1] BOYER R S, MOORE J S. A fast string searching algorithm[A]. Communications of the ACM 20[C]. 1977.762-772.

[2] WU S, MANBER U. A Fast Algorithm for Multi-Pattern Searching[R]. Technical Report TR-94-17, University of Arizona at Tuscon, 1994.

[3] AHO A V, CORASICK M J. Efficient string matching: an aid to bibliographic search[A]. The Communications of the ACM[C]. 1975. 333-343.

[4] COMMENTZ-WALTER B. A string matching algorithm fast on the average[A]. Proc 6th International Colloquium on Automata, Languages, and Programming[C]. 1979. 118-132.

[5] TUCK N, SHERWOOD T, CALDER B, *et al*. Deterministic memory efficient string matching algorithms for intrusion detection[A]. Proc of the IEEE Infocom04[C]. Piscataway: IEEE, 2004. 333-340.

[6] WANG Y C, SHEN Z, XU Y Z. Improved algorithms for matching multiple patterns[J]. Journal of Computer Research and Development, 2002,39(1):55-60.

[7] LI W N, EY P, QIAN H L. Multi-pattern matching algorithms and hardware based implementation[J]. Journal of Software, 2006, 17(12): 2403-2415.

[8] SHEN Z, WANG Y C, XU Y Z. A fast multiple pattern algorithm for Chinese string matching[J]. Journal of Software, 2008, 19(3): 674-686.

[9] SUN Q D, HUANG X B, WANG Q. Multiple pattern matching on chinese/english mixed texts[J]. Journal of Shanghai Jiaotong University, 2001, 35(9): 1286-1289.

[10] WONG K F. String matching on Chinese/English mixed texts[J]. Int'l Journal of Computer Processing of Chinese and Oriental Languages, 1996, 10(1): 115-126.

[11] YIN L H, FANG B X. An improved algorithm for multiple patterns matching[J]. Journal of Huazhong University of Science and Technology, 2005, 33(1): 300-303.

（下转第 44 页）

[14] SEUNG JUN B, GUSTAVO DE V, XUN S. Minimizing energy consumption in large-scale sensor networks through distributed data compression and hierarchical aggregation[J]. IEEE Journal on Selected Areas in Communications, 2004, 22(6): 1130-1140.

[15] ANANDKUMAR A, LANG T, SWAMI A, *et al*. Minimum cost data aggregation with localized processing for statistical inference[A]. The 27th IEEE Conference on Computer Communications (INFOCOM'08)[C]. Phoenix, AZ, USA, 2008. 780-788.

[16] YOUNGCHUL S, SASWAT M, LANG T, *et al*. Cooperative routing for distributed detection in large sensor networks[J]. IEEE Journal on Selected Areas in Communications, 2007, 25(2): 471-483.

[17] HEINZELMAN W B, CHANDRAKASAN A P, BALAKRISHNAN H. An application-specific protocol architecture for wireless microsensor networks[J]. IEEE Transactions on Wireless Communications, 2002, 1(4): 660-670.

[18] YAN W, FAHMY S, SHROFF N B. On the construction of a maximum-lifetime data gathering tree in sensor networks: NP-completeness and approximation algorithm[A]. The 27th IEEE Conference on Computer Communications (INFOCOM'08)[C]. Phoenix, AZ, USA, 2008. 356-360.

[19] BAVISH B. Formulations and algorithms for the capacitated minimal directed tree problem[J]. Journal of the ACM, 1983, 30: 118-132.

[20] CORMEN T H, LEISERSON C, RIVEST R L, *et al*. Introduction to Algorithms[M]. Cambridge, MA: MIT Press, 2001.

[21] LIN F Y S, YEAN-FU W. Multi-sink data aggregation routing and scheduling with dynamic radii in WSNs[J]. IEEE Communications Letters, 2006, 10(10): 692-694.

[22] MADAN R, LALL S. Distributed algorithms for maximum lifetime routing in wireless sensor networks[J]. IEEE Transactions on Wireless Communications, 2006, 5(8): 2185-2193.

[23] KIM S, AHN H. Convergence of a generalized subgradient method for nondifferentiable convex optimization[J]. Mathematical Programming, 1991, 50(1-3):75-80.

[24] LUENBERGER D, YE Y. Linear and Nonlinear Programming[M]. 3rd edition. New York: Springer-Verlag, 2008.

作者简介：

唐伟（1980-），男，四川成都人，电子科技大学博士生，主要研究方向为无线多跳网路由算法。

郭伟（1964-），男，四川达州人，电子科技大学教授、博士生导师，主要研究方向为移动通信网、信号与信息处理。

（上接第 36 页）

[12] WANG G G, QIN Z G. Improved AC-BM algorithm for matching multiple strings[J]. Journal of University of Electronic Science and Technology of China, 2006, 35(4): 531-533.

[13] MUTH R, MANBER U. Approximate multiple string search[A]. Proc of the 7th Annual Combinatorial Pattern Matching Symp[C]. Berlin: Springer-Verlag, 1996. 75-86.

[14] GONZALO N, RICARDO B Y. Fast multidimensional approximate pattern matching[A]. Proc of the CPM'99. LNCS 1645[C]. Springer-Verlag,1999. 243-257.

[15] RICARDO B Y, REGNIER M. Fast algorithms for two dimensional and multiple pattern matching[A]. Proc of the SWAT'90. LNCS 447[C]. Springer-Verlag, 1990. 332-347.

[16] GONZALO N. Multiple approximate string matching by counting[A]. Proc of the WSP'97[C].. Carleton University Press, 1997. 125-139.

[17] RICARDO B Y, GONZALO N. New and faster filters for multiple approximate string matching[J]. Random Structures and Algorithms, 2002, 20(1): 23-49.

[18] GAO P, ZHANG D Y, SUN Q D, *et al*. A multiple approximate string matching algorithm of network information audit system[J]. Journal of Software, 2004, 15(7): 1074-1080.

作者简介：

廖唯荣（1978-），男，湖南怀化人，北京大学博士生，主要研究方向为互联网与信息安全。

邹维（1964-），男，重庆人，北京大学计算机科学与技术研究所研究员、博士生导师，主要研究方向为互联网与信息安全。