International Journal of Security and Its Applications
Vol.8, No.3 (2014), pp. 339-354
http://dx.doi.org/10.14257/ijsia.2014.8.3.34

# An Improved Wu-Manber Multiple Patterns Matching Algorithm for Mobile Internet Content Security Audit in a Chinese Environment

Zhenjiang Zhang[1]*, Xinlei Jin[1], Ziqi Hao[1] and Xiaolan Guan[2]

*[1*]Department of Electronic and Information Engineering, Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education, Beijing Jiaotong University, Beijing, 100044, China*
*E-mail: zhjzhang1@bjtu.edu.cn; 11120100@bjtu.edu.cn; 13120069@bjtu.edu.cn*
*[2]Beijing Institute of Graphic Communication, Beijing 102600, China; 08113101@bjtu.edu.cn*

***Abstract***

*The rapid progress of mobile Internet has brought about wide range of applications of security audit systems. As a part of security audit system, content security audit is crucial for ensuring the reliability and security for system applications. In this paper, we propose a novel multiple patterns matching algorithm WMMA based on the Chinese context (Wu-Manber Algorithm for Mobile Internet Security Audit) to realize auditing the interactive content in the mobile Internet. Our improved algorithm based on the efficient WM (Wu-Manber) multiple patterns matching algorithm, which can improve the insufficient of the WM when dealing with Chinese context. The simulation results showed that the improved algorithm is more suitable for the mobile Internet content security auditing of Chinese language environment, which has higher operation efficiency.*

***Keywords:*** *mobile Internet; security audits; WM Algorithm; pattern matching; Chinese character frequency*

## 1. Introduction

Along with the development of mobile Internet, the security issues are drawing increasing attention. As a result, the content security for content-dominant mobile Internet becomes a major problem to be researched.

The existing technologies for ensuring the safety of the network, such as IDS (Intrusion Detect System) and firewall are mainly used to defend and test the external attacks in the traditional Internet environment. However, regarding the mobile Internet, security threat may come from the information content of a mobile application or website; thus, the traditional Internet monitoring methods are not suitable for the mobile Internet. Therefore, it is important to conduct research on monitoring schemes to improve mobile Internet content security [1].

In the field of the mobile Internet content security audit, the mainstream research methods tend to apply pattern-matching algorithms to achieve the function of content filtering for content audit. The pattern-matching algorithm can be defined by finding the specific pattern P in given text T, where T and P are all strings. A match occurs each time P is found in T. The definition assumes $T = T_1T_2 \cdots T_n$ (length is n), the pattern $P = P_1P_2 \cdots P_m$ (length is m), and $m \le n$ . Pattern matching involves searching all locations of P in text T [2]. Multiple patterns matching procedure involves finding all positions of more than one pattern from the given text [3].

WM (Wu-Manber) multiple patterns matching algorithm adopts the idea of the hash technology and high efficiency filtration, which has the advantage of easy processing, logic clarity and operating efficiency. However, the existing WM algorithm-based methods for a achieving pattern matching for monitoring content security are all aim of English language texts. Caused by the inherent characteristics of the Chinese context, the wastage of calculate and storage resources may be increased when using the WM algorithm in Chinese language environment directly. Consequently, it is necessary to propose a method that is suitable for improving the mobile Internet content security within the Chinese content.

Considering the weakness of traditional security ensuring technologies, this paper introduces patterns matching algorithm into mobile Internet, proposing a novel multiple patterns matching algorithm (WMMA) suitable for the Chinese context based on the classical WM algorithm. The proposed algorithm can achieve the goal of auditing for mobile Internet content security. By recording and examining information content from mobile users, the security monitoring system can perform real-time monitoring of content security through performing WMMA and then filter suspicious information. Meanwhile, the system will record the source of suspicious data or information as evidence for follow-up investigation.

The rest of the paper is organized as follows. We present our related work, including the existing Internet content security audit schemes and commonly used pattern-matching algorithms in Section 2. Section 3 briefly introduces WM algorithm, including its basic theory and disadvantages in the Chinese language environment. Section 4 introduces in detail the proposed improved multiple patterns-matching algorithm for mobile Internet content security audit (WMMA). In Section 5, we evaluate the performance of the two algorithms by simulation. In Section 6, we conclude the paper and provide our future work.

## 2. Related Work

The existing network security audit products at home and abroad are basically based on the traditional Internet, most of which only perform audit of the user behaviors without considering specific content from users. Therefore, it is not suitable to apply the existing security audit technologies directly in the content-dominant mobile Internet.

In the field of content security audit, the current mainstream approach applies pattern-matching methods to conduct the auditing and filtering of the content of network applications. Pattern-matching algorithm can be divided into two categories, single pattern matching algorithm and multiple patterns matching algorithm. The former involves only one pattern exists while the latter involves matching multiple patterns at the same time. BF (Brute Force) [4] is the most primitive single pattern-matching algorithm, the basic idea of which is to compare the pattern to the text one by one from their first character, if the first pair of characters is equal, and then the next pair is examined. Or the matching will turn to comparison between the second character of target text and the first character of the pattern, and so on. It is obvious that BF is a quite simple algorithm; however, it is inefficiency for brute force.

The single pattern-matching algorithm can be classified into three types, the prefix-based searching method, the suffix-based searching method, and substring-based method, according to the searching order. KMP (Knuth-Moms-Pratt) [5] is one of the most classical prefix-based searching algorithms. It performs the matching from first character of pattern, and the maximum shift of pattern depends on the correctly matched prefix. Boyer-Moore proposed in [6] is a represent of suffix-based searching algorithms, which search from the last character, and the maximum shift of pattern depends on the maximum value of independent shift distances of two shifting rules called good prefix and bad character, respectively. One typical substring-based method is BDM (Backward-Dawg-Matching) [7]. It adopts a complex data structure based on suffix automaton. Over the single pattern-matching algorithms, BM could reach the linear time complexity in the worst case to achieve nearly sub-linear time complexity on average. In most cases, BM is superior to the other single pattern matching algorithms and is widely used. However, BM has many disadvantages. When the pattern is quite short, most shifts use bad character rule while good prefix rule is rarely used; thus, the advantages of BM will not be highlighted. Horspool [8] algorithm improved the BM, making it more efficient when the pattern is relatively short. The authors in [9] proposed an improved BM algorithm and applied it to the engine of network intrusion detection system, which is very useful for enhancing the overall capacity. [10] proposed a series of pattern matching algorithms that are more effective compared to BM. These algorithms can be divided into three types, including a quick search algorithm, a maximal shift algorithm, and an optimal mismatch algorithm.

Multiple patterns matching algorithm differs from single pattern matching algorithm in that the former can search more than one pattern in the text at the same time. AC (Aho-Corasick) [11] and WM(Wu-Manber) [12] algorithms are two widely used multiple patterns matching algorithms. AC is the extension of KMP applied to multiple patterns. This algorithm adopts a special automaton named Aho-Corasick automaton, which is generated by the patterns. Meanwhile, it adopts a specific data structure TIRE to perform the pretreatment of patterns, which can reduce time consumption in single pattern matching. However, there are also some shortcomings of AC, such as the lack of jumping rule for the characters unnecessary to be compared. WM algorithm evolves from BM algorithm, so it maintains the matching order of from back to front of BM and also determines the next match by good prefix rule and bad character rule. Compared with AC, WM is faster, simple, and easier to realize. Thus, in this paper, we use WM algorithm to make further improvement. Many improved WM algorithms [13-16] have been proposed to improve the performance of the original WM algorithm. The authors in [13] present an improved WM algorithm, which divided patterns into different sets according to their length. A specific matching method and data structure will be used for each set and searching will be performed simultaneously for different sets. By this means, the problem of low efficiency caused shorter patterns will be solved. An improved WM algorithm DHSWM [14] was designed to resolve the problem of the less efficiency with the constant increase of number of patterns. [15] proposed an improved WU-MANBER algorithm used in the intrusion detection, the matching number of the proposed algorithm is much fewer than AC-BNFA algorithm.

All of the above-mentioned pattern-matching algorithms can be used to realize the mobile Internet content audit. However, the existing algorithms are mainly based on the English characters, and the matching efficiency in this context is unsatisfactory for Chinese language. This paper adopts multiple patterns matching algorithm to design mobile Internet content security auditing scheme. In view of the shortcoming of traditional matching methods when dealing with Chinese text, we put forward a novel effective Chinese pattern-matching algorithm based on WM algorithm.

## 3. Introduction to WM Algorithm

This paper presents the mobile Internet content security audit program based on the WM (Wu-Manber) multi-pattern matching algorithm. The following section briefly describes the WM algorithm.

### 3.1. Overview of WM Algorithm

WM multi pattern matching algorithm applies hash method by pre-processing of model train set, creating three tables, SHIFT, HASH, and PREFIX. SHIFT is used to decide the shifting distance when mismatch happens during matching. HASH and PREFIX are used to decide which specific pattern needs to be matched when the SHIFT Table has a successful match.

SHIFT table: Considering the size of the character block B, rather than simply a character, block transfer characters are used. Generally B=2 or 3, SHIFT build an index for all the possible characters the length of which is B. Therefore, the size of SHIFT is the possible permutations of B characters. The SHIFT value decides the moving distance of a string of some certain B-characters in the text, namely the distance between the most right the certain B-characters and the tail of all patterns. Suppose X is B-length character block of the current calculation and its hash value is i, Consider two cases: First, X does not appear in any of the string pattern, string matching algorithm of the current text moves distance m-B +1 characters position, so we store m-B +1 in SHIFT [i]. Second, X string appears in some modes, in this case, the algorithm matches the rightmost position X that appears in the pattern string.

Assuming X in P [j] appears in the position q, X appearing in the other modes in the X position of the string is not greater than q. Thus, we should store m-q in SHIFT [i].

This is expressed by the formula:

$$SHIFT[i] = \begin{cases} m - B + 1, X \notin \Sigma \\ \min\{m - j \mid X_{[k]} = P_i[j - b + k], 1 \le k \le B\}, X \in \Sigma \end{cases} \tag{1}$$

Finally, we will get SHIFT table, the values stored in the table represent the maximum safe distance of a long string of B when we are able to move the text. Checking the position pos and obtain the hash value i of the block B, when the SHIFT [i]> 0, pos = pos + SHIFT, move backward, as shown in Figure 1.

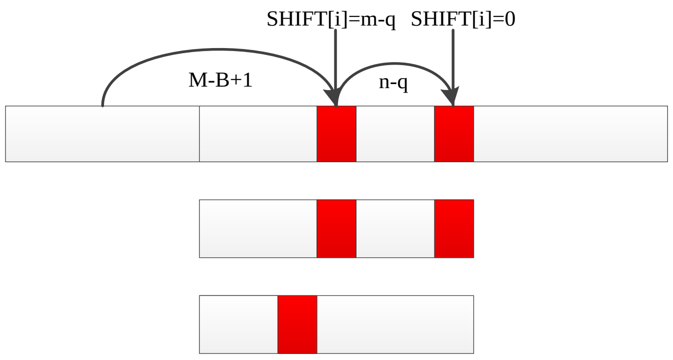

**Figure 1. WM Algorithm Jump Figure**

**HASH table:** As SHIFT [i] = 0, the representative of the current position of the X-match series on the tail might match a certain mode. Hence, HASH [i] points to the long tail of B block hash value of the character mode i head of the list p. We can sort and store all hash values of the last B-character block of the patterns in a pattern array, in ascending order of moving pointer p we can find all the patterns have the hash values of i, and p = hash [i +1] represents the tail of the linked list.

**PREFIX table:** When SHIFT [i] = 0 and HASH table lists all the possible modes when using the string. By performing a hash head B 'character, hash value of each mode appears in the PREFIX table. HASH [i] also points to the PREFIX table, the hash value of the character is determined by comparing PREFIX [p] and the first matching string B ', which can be further determined through the pattern matching. Eventually, the pattern string and each character of the text string are compared to determine whether they match.

If SHIFT [i] = 0 and the match is completed, pos = pos +1, continue to check SHIFT for the location pointer pos.

Specific matching process is as follows (text, said text string, pat said pattern string):

**Step1.** Compute the hash value h of the last B-bit of the text scanned through the hash function;

**Step2.** Checking the SHIFT table, if SHIFT [h]> 0, then the text pointer pos slip a distance of SHIFT [h], go to step 1; if SHIFT [h] = 0, go to step 3;

**Step3.** Calculate the hash value of this m-bit prefix of current window, denoted by text_prefix;

**Step4.** For each p ( $HASH[h] \le p < HASH[h+1]$ ), see whether PREFIX [p] = text_prefix. If they are equal, then select this pattern to match the current text window.

## 3.2. WM Algorithm Performance Analysis

Practice has proved that most of the time, SHIFT is not zero. In a typical experiment, when the quantity of the patterns is 100, SHIFT equals zero 5 percent of the time, when the quantity is 1000, SHIFT equals zero 27 percent of the time, and when quantity is5000,SHIFT is zero 53% of the time [17],which means that matching string is jumping ahead so it can achieve sub-linear time complexity. After calculation, the computation complexity is $O(mp) + O(BN / m)$, where N is the size of the text, P is the number of patterns, and m is the length of each pattern.

WM algorithm requires that all patterns have nearly the same length, which should not be too short because the maximum distance of SHIFT table is limited by the minimum length of the patterns. If the shortest string is too small, then the displacement distance will be quite small as well, seriously influencing the matching efficiency. However, compared with the AC algorithm, WM algorithm is fast, has simple data structure, is easy to implement, and has much better performance compared with AC algorithm; hence, this paper chose WM multi-pattern matching algorithm to improve perform content security audit of mobile Internet.

### 3.3. Insufficient of WM for Chinese Context

WM algorithm combines BM algorithm jump ideas and hash knowledge. Although it has done a lot to improve the efficiency of the multi-pattern matching, there are still some flaws and room for improvement in the Chinese context.

First, in the Chinese context, the pattern string length characters are very short, but the WM algorithm will calculate the hash value of last B characters of current matching window regardless of the length of the pattern and creates the jump table by querying SHIFT. If the pattern string length is very small (for the time being, assuming m = 2), the maximum displacement obtained from m-B +1 is 1, indicating that hash value should be calculated for every move, which takes too much time, seriously affecting the efficiency of the overall matching.

Second, PREFIX table in WM is applied to distinguish the pattern strings with the same suffix, but in Chinese matching case, due to the complex characters, the chance of more than two patterns sharing the same suffix is extremely small, weakening the advantage of PREFIX table. In Chinese text matching, each time a query of PREFIX table is performed will definitely increase the number of unnecessary memory accesses and spend a lot of extra time, thus lead to matching efficiency reducing. Meanwhile, using the PREFIX table also consumes many system resources.

Finally, matching the Chinese text is different from matching English characters. Each B character block is meaningful, unlike English language, of which just a few letters has no real meaning. The particularity of Chinese characters can be applied to improve the sequence of matching. Moreover, when the pattern match fails, the shifting distance of WM algorithm can be improved as well.

From the above, this paper improved the deficiencies of WM algorithm in Chinese case, including eliminating the prefix table PREFIX which saving the corresponding query time and storage space. Additionally, we introduced a new TRAIL table for relatively short patterns of Chinese string. By the trail, we can reduce the time of hash calculations and avoid excessive hash calculation caused by short patterns. Moreover, Chinese character frequency model was introduced in this paper and TF-IDF keyword weights were used to decide matching order for multi-pattern matching, improving the operational efficiency. The improved algorithm is more suitable for mobile Internet content security audit system of Chinese context.

## 4. The Improved WM Algorithm

### 4.1. A Scientific Measurement for the Weight of Keywords TF-IDF

We first illustrate the concept of the TF-IDF, which is a scientific measurement of the weight of keywords [18]. TF - IDF is recognized as the most important innovation in the field of information retrieval. It is widely used in the existing search, literature classification, and other related areas. In this paper, we apply it to calculate the probability of the keywords to improve the accuracy of the words correlation, which is called the probability of useful occurrence.

We use an example to facilitate the understanding of the concept. The patternstring“原子能的应用” can be divided into three keywords: “原子能”、“的”、“应用”. Based on the intuition, we know that the texts that contain these keywords more are more related to the pattern than the text that contains less. Of course, this method has an obvious flaw. The long texts have advantages over the short ones, as the long texts contain more keywords. Therefore, we need to normalize the number of key words according to the length of the text by dividing the number of keyword occurrences by the total number of words in the text. We call this result the Term Frequency. For example, there are 1000 words in one text, and the number of the keywords occurrences that include“原子能”, “的”, “应用” is 2, 35, and 5. Then, Term Frequency of these three keywords is 0.002、0.035, and 0.005. The result of adding these three values together is the Term Frequency of the patternstring “原子能的应用” for the corresponding text.

Therefore, we can use the Term Frequency to present the correlation between the text and the keywords. For example, if there are N keywords $w_1, w_2, \cdots w_N$, and the Term Frequencies in one text are $TF_1, TF_2, \cdots TF_N$. Then, the correlation of this pattern string is

$$TF_1 + TF_2 + \cdots + TF_N \tag{2}$$

However, there is a flaw. In the above example, the "的" accounts for about 50% of the total word frequency, and it is almost useless to determine the theme of the text. We call these words "stop words", that is to say, we should not consider their frequency when measuring correlation. In Chinese language, there are many stop words, such as ″是″、″和″、″中″、″地″、″得″, and so on. Ignoring the stop words, the correlation of pattern string above becomes 0.007.

However, there is another flaw. In Chinese, ″应用″ is a very common word, and ″原子能″ is a very professional word. Therefore, the relevancy of those professional and specific words is more important than that of the common words. We then need to assign a weight to each word. The weights must meet the following two conditions:

(1) The more relevant between the keywords and the theme, the greater the corresponding weight will be, otherwise the weight will be smaller.

We can more or less understand the theme of a text string when we see the word ″原子能″. However, we cannot understand the theme of a text string when we see the ″应用″. Therefore, the weight of ″原子能″ should be larger than ″应用″.

(2) The weight of the stop words should be zero.

This is easy to understand, if a keyword only appears on a few pages, it could be found easily, and so it should be the assigned larger weight. On the other hand, if a word appears in large amount of texts, however it still hard to clear the real meaning of the word, so it should be assigned a smaller weight.

When calculating the weight, assume that one keyword w appears in $D_w$ texts, then the larger the $D_w$, the larger the weight of w, and vice versa. In the multi-pattern matching algorithm, we use the method of Inverse Document Frequency (IDF) to get the weight. The formula is $\log(D / D_w)$, where D is the total number of the text strings, if $D_w = 1000$, then the value of the weight IDF=log（1000/1000）=log（1）=0. If the ″原子能″ appears in 2 text strings, then $D_w = 2$, the weight IDF=log(500)=8.96. If the ″应用″ appears in 500 text strings, its weight IDF=log (2) =1. Using the IDF, the correlation calculation formula changes from simple summation of word frequency to the weighted summation, namely

$$TF_1 \times IDF_1 + TF_2 \times IDF_2 + \cdots + TF_N \times IDF_N \tag{3}$$

In the example above, the correlation of ″原子能的应用″ between the text string and the pattern string is 0.0161, in which the ″原子能″ contributes 0.0126, and ″应用″ only contributes 0.0035. This proportion is more consistent with our intuition.

## 4.2. Description of the WMMA Algorithm

In our proposed WMMA algorithm, the weight of the keywords matching would take priority. Here, the weight of the keywords reflects the correlation between the theme of the pattern string and the characters block B described in the WM algorithm. Because not every block is related to the theme, we first match the block B which has higher correlation, to reduce the number of matching times and improve the efficiency of matching.

We will illustrate the WMMA algorithm in detail in the following.

First, cancel the prefix table. As we have said before, in Chinese environment, repeated frequency of the suffix characters block is low. If the prefix table is reserved, it will not only waste the system storage space, but also waste the matching and reduce the efficiency, as the prefix must be scan several times in the matching phase.

Second, because the length of the Chinese character pattern string is short, generally 2 ~ 5 Chinese characters, the WMMA algorithm introduces a suffix table TRAIL to reduce the number of hash calculation when the shortest length of pattern string is 2. This table is established using the index, which is based on last characters of the shortest pattern string. The benefits of introducing TRAIL table is that the secure mobile distance will be greater than or equal to the original WM algorithm, regardless of whether the matching is success.

Then, not according to the original WM matching algorithm that matches characters one by one the from right to left, WMMA will first obtain the keyword weight using TF – IDF and the matching order depends on the weight of block character. The greater the weight is, the more preferential matching order will be. This method reduces the number of matching times and improves the efficiency of matching.

Finally, in WMMA algorithm, when the length of block B is 2, we just need to analyze the last character when examining the suffix of the characters block. The SHIFT table decides the moving distance of current matching window according to characters' block, which comprises the last character of the current match window and the adjacent next character. Here, the calculation method of jump distance is different from the WM algorithm, which will be described in the following. This method can provide greater moving distance compared to the WM algorithm and improve the efficiency of the match. WMMA algorithm will take HASH table as a further entrance of comparison, and each moving distance is not related with whether the HASH value is zero or not.

The flow chart of WMMA algorithm is shown in Figure 2.

The following is the flow of WMMA algorithm:

The WMMA algorithm needs to generate 3 tables in the preprocessing phase, including the SHIFT, HASH tables and the TRAIL table, without the PREFIX table, which is different from the WM algorithm. The computing and storage of TF-IDF, which is the measurement of the weight of key words is also finished in this phase. The calculation of TF-IDF for pattern string set will be performed only once before matching and the results could be used all the time, which is a conservation of resources.

First, we should get the shortest length min_len of the pattern strings, setting the length of the character block B=2. Subsequently, we compute hash value of the first min_len length of the string and the last B characters to obtain a hash table. The hash table is set up just like the WM algorithm.

When setting the SHIFT table, the max moving distance is not computed the same as the WM algorithm. The WMMA algorithm merges the last character of the current matching window with the first character of the next window to get a block with the length of 2 and compare it with the pattern string. The computing process is as follows.

Assuming the block is B[0~1], B[0] is the last character of the current window and B[1] is the first character of the next window. We can get the hash values of B[0~1] and use them to access the shift table. The result represents the distance that the block can move.

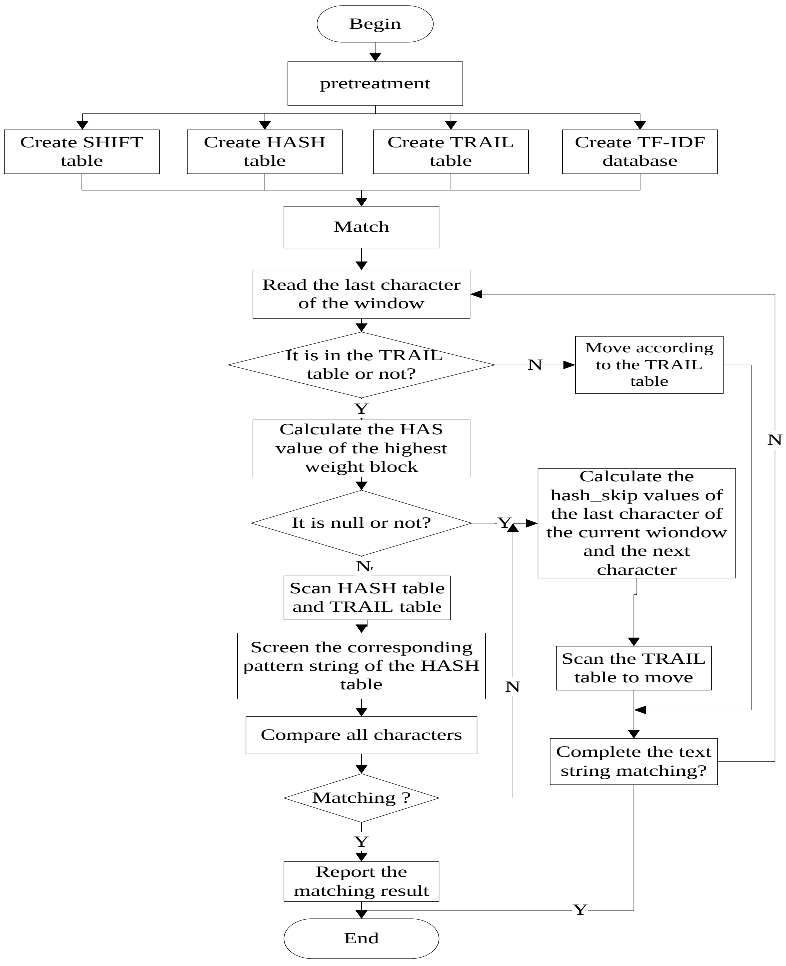
Begin
pretreatment
Create SHIFT table
Create HASH table
Create TRAIL table
Create TF-IDF database
Match
Read the last character of the window
It is in the TRAIL table or not?
N
Move according to the TRAIL table
Y
Calculate the HAS value of the highest weight block
N
It is null or not?
Y
Calculate the hash_skip values of the last character of the current wiondow and the next character
N
Scan HASH table and TRAIL table
Screen the corresponding pattern string of the HASH table
Scan the TRAIL table to move
N
Compare all characters
Complete the text string matching?
Matching ?
Y
Report the matching result
Y
End

**Figure 2. WMMA Algorithm Flow Chart**

Supposing the hash value of B[0~1] is represented by hash. Divide the matching circumstances into the following three situations.

(1) If B[0~1] appears in the first min_len characters of some patterns, and the location of the most right of B[0~1] is j, then the safe move distance is min_len-j-1, that is, SHIFT[hash]=min_len-j-1.

(2) If B[0~1] never appears in any of the pattern strings, however the suffix B[1] is the prefix (first character of patterns) of some pattern strings, then the safe move distance is SHIFT[hash]=min_len.

(3) If B[0~1] never appears in any of the pattern strings and the suffix B[1] does not emerge in any of the prefixes of any pattern strings either, then the safe move distance is SHIFT[hash]=min_len+1.

The next step involves establishment of the TRAIL table. The TRAIL table is indexed by the last character of the unified-length patterns, and the order of TRAIL table also follows the frequency of Chinese characters. We can build the frequency table of the characters using the BMMA algorithm. We definite the move distance of TRAIL table as follows: when the last character does not appear in any word, its move distance equals to min_len; when it appears, its move distance equals to min{d( $p_j p_{\min-len}$ ), min_len}.

The last step is to build the TF-IDF library. First, we should extract key words of relevant pattern strings from 5000 random text samples. Second, obtain the weight for key words as the method mentioned in section 4.1. Finally, put the weight into the TF-IDF library. During pretreatment, we should find a correspondence option for pattern matching string collections and sort the character block by weight.

Thus, the pre-processing stage of WMMA algorithm is completed.

In the matching stage, due to the WMMA algorithm block character set length B = 2, it is assumed that the character block B [0] and B [1] is in the current matching window. Assume that C is the character next to the matching window, we get the hash called hash_skip from B[0] and B[1].

When doing matching, we should find whether the last character in the current window is also in the TRAIL table. If not, we do not need to calculate hash value, just perform moving according to the TRAIL table. If the last character is also in the TRAIL table, we should compute the hash. Different from the left to right in WM algorithm, the WMMA prioritize matching character block that has the maximum weight, and then take the hash table for further comparison. If the HASH[hash] is empty, it means the block composed by B[0] and B[1] does not match the character blocks that have the highest weight. If not, it means the block composed by B[0] and B[1] matches the block has the maximum weight of some patterns. At this time, the hash table comprises the list of the number of pattern strings that may be matched, and then to find the last character in the TRAIL table that matched (here the function of TRAIL table seems like PREFIX in WM), and if we find a match, we record the relevant information. After judging the HASH[hash], move the current matching window by a distance of hash_skip without considering whether the HASH[hash] is empty or not.

The main steps of matching stage for WMMA algorithm are as follows:

1. Analyzing whether the last character which is right-alignment to the current window of text string is in the table TRAIL. If so, go to step2; if not, perform moving according to TRAIL table;
2. Calculating hash values of the block characters that have the highest correlation weight of the current window;
3. Judging whether the HASH [hash] is empty, if empty, go to step 5, otherwise continue;
4. Perform further matching of the selected patterns in step 3. If one of them is completed matching with current window, then we record relevant information;
5. Computing the hash value *hash_skip* of the block composed by the last character B[1] in the current window and the next character C, and then moving a distance of SHIFT[hash_skip]. Then to judge whether the pattern string is finished, if not, go to step 1 to perform next round of matching. Table 1 shows the pseudo-code of WMMA algorithm.

**Table 1. The Pseudo-code of Algorithm WMMA**

```
WMMA（$P = \{p^1, p^2, \cdots, p^r\}$， $T = t_1 t_2 \cdots t_n$）
1.  Preprocessing
2.      Computation of  B
3.      Construction of the hash tables SHIFT and HASH
4.      Construction of the table TRAIL
5.      Search for the TF-IDF Base
6.  Searching
7.   pos=len_min, len[B]=2, B[last]
8.  While pos≤ n
9.   Do hash_skip=B[last]B[last+1]
10. If B[last]∈ table TRAIL
11.  Then computing HASH[hash] of the highest weight B
12.  Elsepos=pos+min{ d（$p_j p_{\min-len}$）， min_len}
13. End of if
14. If HASH[hash]≠ $\phi$
15.  Then searching tables HASH and TRAIL
```

```
16.           Verify all the patterns in list one by one against the text
17.  Else computing SHIFT[hash_skip]
18.  pos=pos+ SHIFT[hash_skip]
19.  End of if
20.  If all the patterns equal the text
21.           Then report an occurrence at pos+1
22.  Else computing SHIFT[hash_skip]
23.  pos=pos+ SHIFT[hash_skip]
24.  End of if
25.    End of while
```

To better understand the WMMA, we give the following example to illustrate the steps of algorithm.

Suppose the text ″制定和完善信息化可以加速国家发展″ and patterns ″互联网″、″信息化″、″信息安全″. At first, we should do preprocessing to generate the SHIFT table, HASH table, and TRAIL table. Assume the length of character block B=2, when the shortest length of the pattern min_len=3, to gain the following TRAIL table according to the using frequency of Chinese characters by national of standard GB2312-80.

**Table 2. TRAIL Chart of the WMMA Algorithm**

| number | trail |
|---|---|
| 1 | 化 |
| 2 | 安 |
| 3 | 网 |

Left justify the text and patterns to start matching, according to WMMA, first, search the last character ″和″ of current matching window in TRAIL table, for the character is not exist, then traverse the patterns set to find ″和″ and the result is negative too, thus shift the matching window by the distance min_len=3 for next round matching. In the second matching time, the last character of matching window is “信”. Thus, we perform query of TRAIL table and conclude ″信″ does not exist in the table, however, it is found in the two patterns: ″信息化″and ″信息安全″. Thus, the shifting distance is 2 according to the shift rule of TRAIL table of which the shift distance is $\min\{d(p_i p_{\min\_len})\}$, where $d(p_j p_{\min-len}) = 2 < \min\_len = 3$. Figure 3 demonstrates the above process.

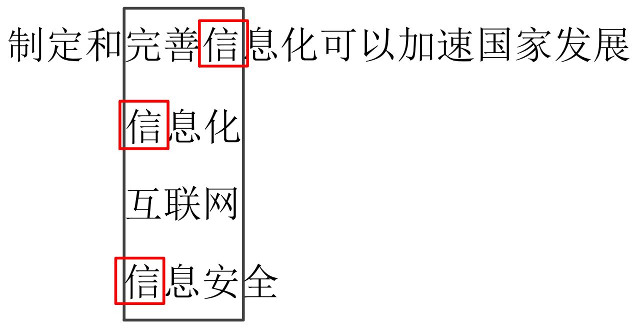

**Figure 3. The Position of the Last Character in the Pattern String**

After shifting, the third round of matching will be performed, as shown in Figure 4. Thelast character of matching window is ″化″, and it can be found in TRAIL table. Next step involves calculating hash value of the character block, which has the largest TF-IDF weights.

We then search this hash value in HASH table gained in preprocessing stage, and the consequence is positive. Finally, we screen out the pattern ″信息化″ as probable matching result. It can be seen that by adopting the TRAIL table, search scope can be reduced.

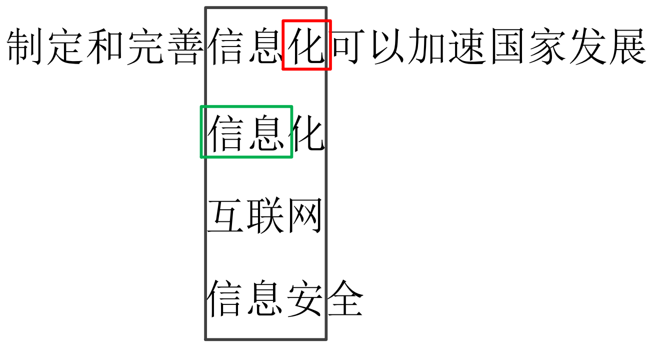

**Figure 4. The Condition of the Last Character Appears in the TRAIL Table**

The last step involves comparing the probable matching result ″信息化″ with current window in target text one character by another. We found the pattern matches the text completely. We then shift the current matching window by a distance of length of the current pattern and start a new round of matching. As in the following rounds, the last characters of matching windows do not exist in TRAIL table. The matching of the target text finishes when it reaches the character ″展″.

## 5. Performance Analysis of Algorithms

### 5.1. Analysis of Computational Complexity

In the pretreatment process, WMMA algorithm introduces a new TAIL table TRAIL. However, the complexity of system space occupation does not become higher due to abandoning the PREFIX table from original WM algorithm. Also, the establishment of TF-IDF library can be completed before matching, which do not consume the time for pattern matching.

The time complexity of the WM algorithm is $O(mp) + O(BN / m)$, where N is the size of the text, P is the number of patterns, and m is the length of each pattern. The time complexity of the WMMA algorithm could achieve $O(mp) + O(BN / m + 1)$ in theory, which is smaller than that of WM. In WM algorithm, because the maximum moving distance in SHIFT table is restricted by pattern string minimum length, so all patterns have to be same in length and no too short patterns is allowed. If the shortest pattern string is quite small, so will be the move distance. As we have to compute the hash value for every movement, which affects the matching efficiency seriously. While, WMMA algorithm has solved the problem of frequent hash value calculation by introducing the TRAIL table. At the same time, the WMMA enlarged the moving distance by improving the SHIFT table, thus to improve the efficiency of algorithm on both sides.

### 5.2. Simulation

The texts strings used in the experiment are comprised of two random text data of which the size are 512K and 1M respectively, and the model train set comprised a collection of respective words. We observed matching time of WM and WMMA algorithm by increasing the number of pattern strings.

The time comparison of WMMA and WM are shown in Figure 5 and Figure 6, of which the text size are 512K and 1M data respectively.

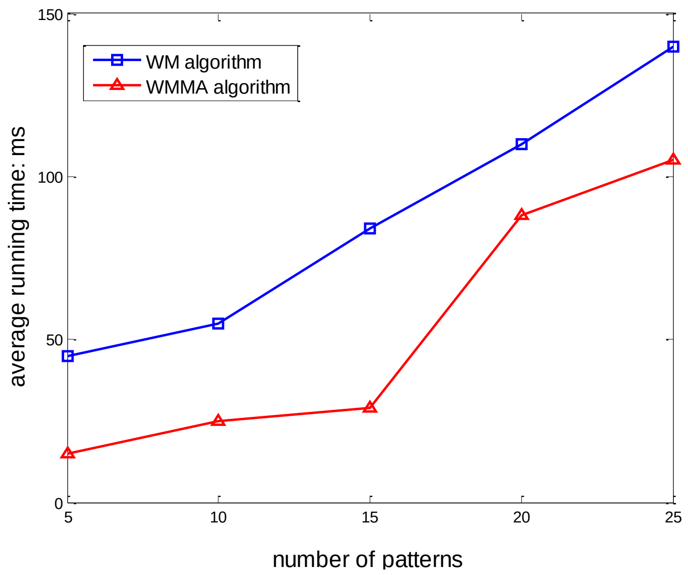

**Figure 5. 512K Text Comparison Chart**

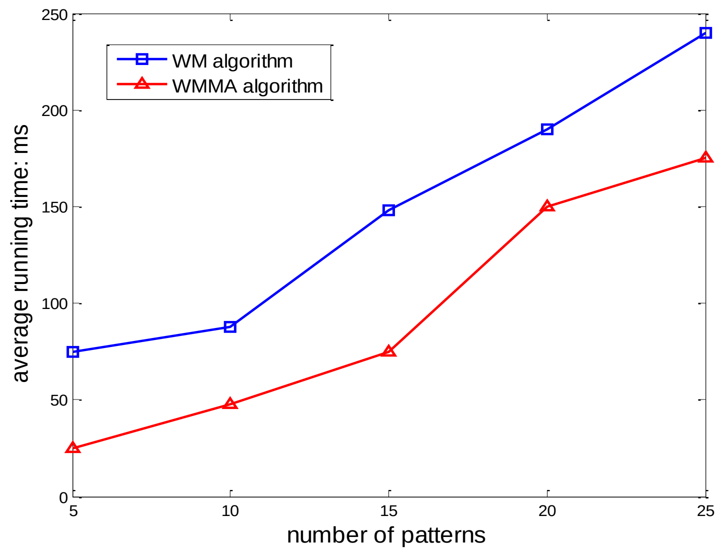

**Figure 6. 1M Text Comparison Chart**

We can see from Figures 5 and Figure 6 that with the increase of the number of pattern strings, the matching times of two algorithms are all increasing. Nevertheless, we can also find that the matching time of WMMA is always shorter than the WM algorithm, thus the efficiency has been significantly improved.

Next, we will observe the performance of WMMA and WM when the text size increases. For example, when the number of pattern strings is 10 and 20 respectively, the comparison of the growth of matching time between the WM algorithm and WMMA algorithm with the text size increases from 512K to 1M is shown in Figure 7.

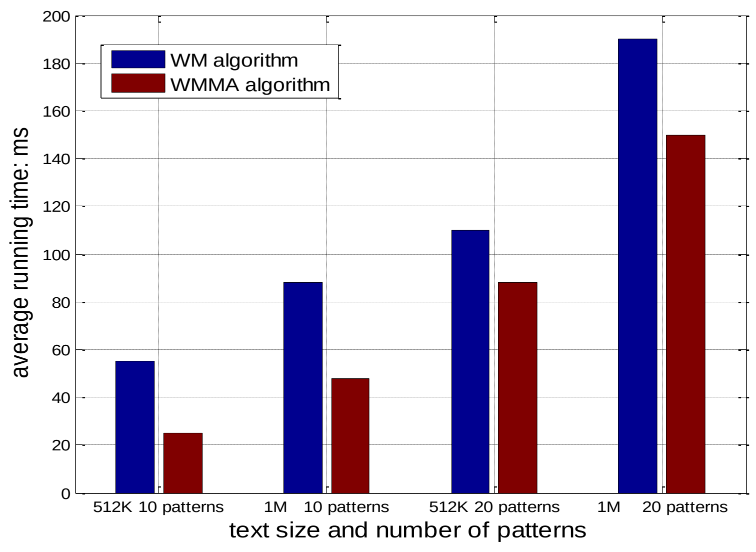

**Figure 7. The Comparison of 512K and 1M**

As can be seen from Figure 7, when text size increases from 512K to 1M with 10 patterns, the running time of WM increased from about 57ms to about 88ms, which increased by about 54%. However, WMMA increased from about 24ms to about 47ms, which increased by about 96%, which seems that the advantage of WMMA have a tendency to decrease when the text size increased. Whatever, in a global view, the running time of WMMA was always shorter than that of WM, so we can still conclude that the efficiency of the WMMA is superior to WM.

From the above, we can draw the following conclusions:

(1)When the text data is the same, with the increase in the number of pattern string, WMMA algorithm's matching time was shorter than the matching time achieved with the original WM algorithm, and the matching efficiency has improved significantly.

(2) In a global view, when the text data changes, the efficiency of the WMMA algorithm was always better compared to WM.

Through these simulations, the proposed WMMA algorithm had better efficiency compared to the original WM algorithm when dealing with the Chinese context.

## 6. Summary and Outlook

In this paper, we studied the basic theory and analytical methods of mobile internet content security audit program by pattern matching algorithms. We studied the advantages and disadvantages of multi-mode matching WM algorithm. Considering the weakness of traditional algorithms under a Chinese language environment, we proposed an efficient algorithm for mobile Internet content security audit, namely the WMMA. Through simulation analysis, the improved WMMA algorithm is more suitable for Chinese locales, which significantly improved the matching efficiency.

Subsequent research work will address the following aspects:

(1) Security audit of the mobile Internet is a complex process that includes not only the audit of illegal content information, but also other aspects such as audit of illegal behaviors and security audit for abnormal data flow. Therefore, in the future work we will consider other aspects of the audit to ensure the mobile Internet security.

(2) The WMMA multi-pattern matching algorithm proposed in this paper concerns mainly the content of Chinese environment audit. If the audit content contains a lot of English text, the improved algorithm is less likely to apply. Thus, in the future the algorithm could be improved to fit pattern matching of content composed by both Chinese and English.

## Acknowledgements

This work was partially supported by the National Natural Science Foundation of China under Grant 61172072, 61271308, the Beijing Natural Science Foundation under Grant 4112045, the Beijing Science and Technology Program under Grant Z121100000312024, the Academic Discipline and Postgraduate Education Project of Beijing Municipal Commission of Education.

## References

[1] D. Xuejuan, "Brief talk of web security audit", Science and Technology Innovation Herald, vol. 28, no. 16, **(2010)**.

[2] F. Hongbo and Y. Nianmin, "A Fast and Exact Single Pattern Matching Algorithm", Journal of Computer Research and Development, vol. 46, no. 8, **(2009)**, pp. 1341-1348.

[3] G. Navarro and M. Raffinot, "Flexible Pattern Matching in Strings", Publishing House of Electronics Industry, BeiJing, **(2007)**.

[4] N. Tuck, T. Sherwood, B. Calder and G. Varghese, "Deterministic memory-efficient string matching algorithms for intrusion detection", Proc of IEEE INFOCOM, Piscataway, **(2004)**.

[5] D. E. Knuth, J. H. Morris and V. R. Pratt, "Fast Pattern Matching in Strings", SIAM Journal on Computer, vol. 6, no. 2, **(1977)**, pp. 323-350.

[6] R. S. Boyer and J. S. Moore, "A fast string searching algorithm", Communications of the ACM, vol. 20, no. 10, **(1977)**, pp. 762-772.

[7] A. C. Yao, "The complexity of pattern matching for a random string", SIAM Journal on Computing, vol. 8, no. 3, **(1979)**, pp. 368-387.

[8] R. N. Horspool, "Practical fast searching in strings. Software: Practice and Experience", vol. 10, no. 6, **(1980)**, pp. 501-506.

[9] S. Wen-jing and Q. Hua, "Improved BM algorithm and Its Application in Network Intrusion Detection", Computer Science, vol. 40, no. 12, **(2013)**, pp. 174-176.

[10] D. M. Sunday, "A very fast substring search algorithm", Communications of the ACM, vol. 33, no. 8, **(1990)**, pp. 132-142.

[11] V. Ahoa and J. Corasickm, “Efficient string matching: an aid to bibliographic search”, Communications of the ACM, vol. 18, no. 6, **(1975)**, pp. 333-340.
[12] S. Wu and U. Manber, “A fast algorithm for multi-pattern searching”, TR 94-1 7. Tucson, AZ: Department of Computer Science, University of Arizona, **(1994)**.
[13] B. Zhang, X. Chen and Z. Wu, “High concurrence Wu-Manber multiple patterns matching algorithm”, Proceedings of the 2009 International Symposium on Information Processing, Huangshan, PR China, **(2009)**.
[14] L. Wei-guo and H. Yong-gang, “DHSWM: An improved multi-pattern matching algorithm based on WM algorithm”, Journal of Central South University (Science and Technology), vol. 42, no. 12, **(2011)**, pp. 3765-3771.
[15] C. Ke-Qin, D. Lin and W. Hui, “An improved multi-pattern matching algorithms in intrusion detection”, Measuring Technology and Mechatronics Automation (ICMTMA), 2013 Fifth International Conference on. IEEE, **(2013)**.
[16] L. Vespa and N. Weng, “SWM: Simplified Wu-Manber for GPU-based Deep Packet Inspection”, Proceedings of the 2012 International Conference on Security and Management, **(2012)**.
[17] Y. Zhang and Y. Li, “Analysis of the Affection of Hash Function on the Performance of WM Algorithm”, 2010 International Conference on E-Product E-Service and E-Entertainment, **(2010)**.
[18] W. Jun, “Beauty of math”, POSTS & TELECOM PRESS, Beijing, **(2012)**.

## Authors

**Zhenjiang Zhang,** is an Associate Professor in School of Electronic and InformationEngineering in Beijing Jiaotong University. He received his PhD degree in TheCommunication and Information System, and his research interests include theWireless Sensor Network and the Computer Communication.

**Xinlei Jin**, received B.E. degree of CommunicationEngineering in Beijing JiaotongUniversity, 2011. Now he is a graduate student in departmentof Electronic and Information Engineering. His major is Communication and Information systems.

**Ziqi Hao**, received B.E. degree of CommunicationEngineering in Beijing JiaotongUniversity, 2013. Now she is a graduate student in departmentof Electronic and Information Engineering. His major is multisource data fusion theories and applications.

**Xiaolan Guan**