Hybrid pattern-matching algorithm based on BM-KMP algorithm

Hou Xian-feng
School of Information Science and Engineering
Changzhou University
Changzhou China
e-mail:hou_xianfeng@126.com

Yan Yu-bao
School of Information Science of Engineering
Changzhou University
Changzhou China
e-mail:yyb12006@sina.com

Xia Lu
School of Information Science and Engineering
Changzhou University
Changzhou China
e-mail:leilei2007_2007@163.com

***Abstract*-- Reading and reference many books and articles, and then analyze and study the KMP algorithm, BM algorithm and a variety of improved algorithms. Summarizes various of the advantages and disadvantages of the pattern matching algorithms and in-depth study the advantages of the algorithms. And on this basis, a new algorithm - KMPBS algorithm is introduced. The algorithm refers to BM algorithm, KMP algorithm and the thinking of improved algorithms. Utilize the last character of the string, the next character and the method to compare from side to side, and then advance a new hybrid pattern matching algorithm. And it adjusted the comparison direction and the order of the comparison to make the maximum moving distance of each time to reduce the pattern matching time.The algorithm reduces the comparison number and greatly reduces the moving number of the pattern and improve the matching efficiency.This algorithm achieved a better result and it works well.**

Keywords: Pattern matching; KMP algorithm; BM algorithm; KMPBS algorithm; hybrid pattern matching

I. INTRODUCTION

The interpretation of string pattern matching is that substring position in the parent string and it is an important algorithm for various systems. More well-known string matching algorithms have BF algorithm, RF algorithm, KMP algorithm and BM algorithm[1,2]etc. Researchers do some more research to improve the algorithm, especially the KMP algorithm and BM algorithm, and the improved algorithm achieved a certain level of matching efficiency[3,4,5]. KMP algorithm and BM algorithm in pattern matching algorithm are to use more times than others. The two algorithms are in the worst case that have linear search time. This paper analyzed and studied to some extent the two algorithms and some improved algorithms. And combination of the two algorithms of thinking and some advantages of the improved algorithm .On this basis, introduced an improved algorithm.

II. CORRELATION ALGORITHM AND IMPROVED ALGORITHM

This text reference to the KMP algorithm and the BM algorithm from left to right balance of contrast and improved method of thinking and some of the advantages of the two algorithms for the following .

A. KMP Algorithm

In the KMP algorithm[1,3]introduced Next function of a pattern string P , for each character of the pattern string P is given a value, use the Next(j) (j = 1 .. m) to illustrate. Next(j) value only depends on the model P itself, the composition of a sequence of characters, nothing to do with the main string. First calculate Next (j) value of the pattern string P, as the following:

$$Next(j) = \begin{cases} 0, & j = 1, \\ \max\{k \mid 1 < k < j \text{ 且 } P[1 \ldots k-1] = P[j-k+1 \ldots j-1]\}, & \\ 1, & \text{else.} \end{cases}$$

The process of string pattern-matching is p [j] (1 <= j <= m) and the text string t [i + j-1] (1<= i <= n-m +1) were compared from left to right : if successful, compared to the next character; if unsuccessful, according to Next(j) function value will be shifted to the right of the string and the current text is in the same position, and i=next(j), then p[j] and t[i] are compared, if the same then compare to the next one, if different, compared to re-determine the location, until the last match to the success or failure.

B. BM Algorithm

BM algorithm is an exact pattern matching algorithms[6,7,8], this algorithm uses right to left comparison of methods, pattern string P along T from left to right and Characters from right to left comparison .After the failure of each comparison, two heuristic methods can be used, that is the bad character rule and the good suffix rule to determine the distance of P to the right moving. BMHS algorithm based on the BM algorithm has been improved[2,8,9], the improved method is that when there does not match the circumstances of the time according to p [m] characters corresponding to t[i+m-1] next character t[i+m] to determine the pattern string P shifted to Right (t[i+m]) of the right amount.

C. The Hybrid Pattern Matching Algorithm

Hybrid pattern matching algorithm is to combine more than one algorithm to form a new algorithm[10,11,12]. The algorithm of this article used in the hybrid pattern matching algorithm is the improved BMHS algorithm. And then make some choices from KMP algorithm combining the improved algorithm --KMPBS algorithm.

KMPBS pattern matching algorithm is also a pattern string P=P[1… m] search from left to right in the text string T=T[1… n]. In the beginning of each step, the starting

978-1-4244-6542-2/$26.00© 2010 IEEE

position of pattern string P=P[1…m] j(j=1) with the position of text string T(T=T[1…n]) i (i=1… n-m+1) Department aligned. Therefore, in the position of m, string pattern character in the text string corresponding to the position (i+m+j). In the KMP algorithm, the pattern string P in a text string T is in accordance with the order from left to right to match, until a successful match or matches to the bad character, and then end of the match or the pattern string P were shifted to the right, if the match to the last character found that pattern matching fails. This will result in front of the wasted work, but according to Next (j) function to obtain the small amount of movement, resulting in more frequent and affect the matching efficiency. BM algorithm is matched from right to left operation, and the KMP algorithm has similar shortcomings.

The idea of the KMPBS algorithm is to maximize the use of string in the last character of the mode with the text string and does not match the corresponding characters in the case of use of end-term character and the next bit character string to determine the maximum distance of the pattern string moved. First, the last character p [m] in pattern string P and the characters of text string in T compared. If successful, KMP algorithm used to match from left to right (for a first character position does not match the pattern string and then get the amount of movement). If it fails, the characters under the text t [i + m-1] and t [i + m] of T to determine the pattern string P of the moving location, while the anchor pattern string pattern string j will return the first character position.

D. KMPBS Algorithm Principle

First of all, they need to preprocess function value and the algorithm used in a number of pretreatment, mainly calculation: Next (j) function of the KMP algorithm, Right (p [j]) values of the BMHS algorithm, and the number of characters in string is classified as a single character, and there appear more than once repeated two cases, and given Single (p [j]) function, the Single (p [j]) function of the role given in the following explained.

Preprocessing, computing the Next (j) function values, Right (p[j]) and Single (p[j]) of the P. Get the value of Right (p [j]) and Single (p [j]) value:

```
void pre(){
for (i = 0; i <ALPHA;i++){
weiyi [i]. right = m +1;
 // for all the characters assigned for the
mode //string length plus one
weiyi [i]. single =- 1;
// for all the characters assigned to -1
}
for (i = 0; i <m ;i++){
weiyi [P[i]]. right = m-i;
if (weiyi [P[i]]. single ==- 1)
weiyi [P[i]]. single = 0;
// mode to a character string appears in a
else if (weiyi [P[i]]. single == 0)
weiyi [P[i]]. single = 1;
// mode to a character string appears twice in
the
else weiyi [P[i]]. single = 1;
// characters appear repeatedly
}}
```

ALPHA is the character table in which the number of characters, default to ASCII character set, 256 elements. When the character is not included in the pattern string then the Right value of this character plus one: m +1; there exist in the pattern string, the assignment for the (m-j): said pattern string length minus the current character string position in the pattern string.

After the above pretreatment, begin pattern matching. Pattern string and text string to match from the left side alignment compared, to the first, p[m] of pattern string and t[i + m-1] (1<= i<=n-m) of the text string are compared. If the same, using the KMP algorithm, pattern string matching process is from left to right starting from the left side compared with the text string, until the match was successful or do not match the first character position and then determine the pattern string shifted to the right amount (determined under the Next function).If different ,the characters Right(t[i+m-1]) and Right(t[i+m]) of the text to determine the pattern string shifted amount to the right. Using the following method to calculate the value of i of the first character position in pattern string corresponding to the location of the text string:

$$i = \begin{cases} i+m+1, right[t[j+m]] == (m+1); \cdots\cdots (1) \\ i+m, right[t[j+m-1]] == m; \cdots\cdots\cdots\cdots (2) \\ i+m+1, \begin{pmatrix} Single(t[j+m-1]) == 0 \;\&\& \\ Single(t[j+m]) == 0 \end{pmatrix} \\ \&\& \begin{pmatrix} (right(t[j+m-1])-1)!= \\ (right(t[j+m])) \end{pmatrix}; \cdots (3) \\ \\ i + ((right(t[j+m-1])-2) > right(t[j+m])\,? \\ (right(t[j+m-1])-1) : right(t[j+m])) \\ \text{except: (1) (2) (3)} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (4) \end{cases}$$

a) If Right(t[i+m]) is equal to (m +1), it shows that this character does not exist in the pattern string, and pattern string P move to the right in length (m +1): such as equation (1). If Right(t[i+m-1]) is equal to m, it is illustrated to show the first character string corresponding to this character, pattern string P shifted to the right amount of m characters long: such as equation(2);

b) If Right(t[i+m]) is not equal to (m +1) or m, in this case would need Single(t[i+m-1]) and Single(t[i+m]) to determine the number of the pattern string shifts to the right along the text string. If the Single(t[i+m-1]) and Single(t[i+m]) is a single character, the algorithm is as follows:

- When the Right (t[i+m-1]) -1 is not equal to Right(t[i+m]), shows character t[i+m-1] and t[i+m] exists, but in the pattern string are separate. So pattern string P, shifted to the right for the distance (m +1): such as equation (3).

- When the Single(t[i+m-1]) and Single(t[i+m]) at most only a single character, then according to the Right(t[i+m-1]) and Right(t[i+m]) to determine the size of the amount of the pattern string shifted to the right : such as equation (4).

The method can be called BMHSI method. In which the modified KMP algorithm:

$$kmp(i) = \begin{cases} i + j - next(j), p[j] != t[i + j]; (1) \\ i, j == m. \quad \cdots\cdots\cdots\cdots\cdots\cdots (2) \end{cases}$$

When j is equal to m, equation (2), the value of i is that match the text string corresponding to the first position in the pattern string after the success. If the character in the process does not match, according to equation (1) pattern string can be moved for the first character in the string of text characters that correspond to the location string.

E. The Main Program of KMPBS Algorithms

After the above steps, you can write on KMPBS function, the following function is KMPBS main parts:

```
int kmpbs (char * p, char * t, int m, int n) {
  for (i = 0; i <n;){
    if (p [m-1] == t [i + m-1 ]){
  // the last character string comparison mode
    kmp (&amp;amp; i);
  // Last character equivalent, use the KMP
  //algorithm to control the match by marking flag
  // With success; if successful, out of circulation,
  //by matching position, or jump to do the
  //following
     if (flag) return i;
       }
    else / / if not equal, use bmhsi algorithm
    bmhsi (&amp;amp; i, m);}
    return i;
  }
```

First of all, use the end-term character of pattern string for pattern matching and use the KMP algorithm to match. The variable of i is the position of current character in the text string, if successful, flag is true (the flag value has to be false, until the success of the assignment is true), if unsuccessful, the i get the new value and then use the BMHSI algorithm to obtain a new position, then loop until the match to send success or failure.

III. ALGORITHM TESTING AND EVALUATION

A. Algorithm Examples

String nearlyfearhotearthepaper is the main string, the string of tear for the pattern for illustration, using two pattern-matching algorithms for comparison, as follows (Table Ⅰ):

Table Ⅰ: KMPBS and KMP algorithms

a.kmpbs algorithm

kmpbs	*nearlyfearhotearthepaper*
Fir time	*tear*
Sec time	*tear*
Thi time	*tear*
Fou time	*tear*
Fiv time	*tear*

b.kmp algorithm

kmp	*nearlyfearhotearthepaper*
Fir time	*tear*
Sec time	*tear*
……	…… ……
……	…… ……
Thirteenth time	tear

KMPBS algorithm: the string of tear with the text near is the same as the last character and then comparison of the first character, not equal (the first cycle compared with the two characters), then use BMHSI function to determine the amount shifted to the right. The next character in the text string for the pattern is not in, therefore, aligned with the yfea and then shifted to the right to use BMHSI function to determine the amount .Text characters a and r in text string shifted to the right, and with fear alignment, tear the last character r the same as the last one with fear. Comparison of the first f and t is different, and use the character h with r to determine the shift length. Tear pattern string does not exist character h, Pattern string with otea alignment, and r as a different character to determine the displacement of one, by comparing the match successfully. Pattern matching is successful, KMPPBS algorithm 10 times, KMP algorithm16 times. The example above is a simple comparison of the two algorithms, KMPBS algorithm is better, the following test is to compare these two algorithms.

B. Algorithm Testing and Evolution

The quality evaluation of an algorithm is from the algorithm's time complexity and space complexity to analyze. The following to compare the two algorithms to compare the number of contrast explanation. The following Table Ⅱ: KMP and KMPBS algorithms compare the number of characters in the form

Table Ⅱ: KMP and KMPBS number of comparisons

Text length	*Pattern length*	*kmp(times)*	*kmpbs(times)*
38	7	53	12
55	13	84	16
267	12	468	35
327	13	628	57

Table Ⅲ: The algorithm of comparing the number of model series mobile

text length	*kmp*	*bm*	*kmpbs*
69	54	29	7
123	108	61	12
177	162	93	18
231	216	125	23
252	168	61	14
267	170	129	25
804	789	367	73

903	888	435	82
1035	1020	586	94

These results from table II can be seen through the time complexity the KMPBS algorithm efficiency has been enhanced than the KMP algorithm that can be reduced more than 60%. However, a certain increase in space spending, need to addressed Right value, Single value, when the text string when the length of this process requires the space and time can be ignored.

KMPBS algorithm greatly reduces the number of times of the movement pattern string, and the KMP algorithm and the BM algorithm were compared in the table (Table III), the length of pattern string is nine characters, gradually increase the length of the text string:

Can be seen the same circumstances from the above table, the new algorithm, the KMP algorithm and the BM algorithm key comparison, moving number has been greatly reduced, KMPHS algorithm mobile number is probably (1/10~1/6) of the KMP algorithm mobile number, the BM algorithm (1/5~1/3), moving significantly reduced the number of the matching efficiency.

IV. CONCLUSION

The results of the above comparison show that the KMPBS algorithm greatly improves the matching efficiency. The most characteristic of the algorithm KMPBS is better use of end-term character and the next bit characters to get the maximum moving distance, and start comparing from the right side and then to compare the left. This approach reduces the last character or the first character does not match the situation and thus reduced the number of comparisons there. The use of pattern matching is very broad and efficient pattern matching algorithm can improve system performance.

ACKNOWLEDGMENT

Many thanks to my teacher for help me to my studies and put forward many ideas and suggestion. And also thanks to my classmates.

REFERENCES

[1] YAN Wei-min, WU Wei-min. Data Structure [M]. Beijing: Tsinghua University Press ,2002,81-84.

[2] Zhang Na, Hou Zheng-feng. A fast improvement of BM pattern matching algorithms [J]. Hefei University of Technology: Natural Science, 2006, 29 (7): 834-838.

[3] Tang Ya-ling. KMP algorithm in the calculation of next array. Computer Technology and Development [J] .2009,19 (6) :98-101.

[4] FRANTISEK FRANEK, CHRISTOPHER G. JENNINGS, WFSMYTH.A simple fast hybrid pattern-matching algorithm [J]. Journal of Discrete Algorithms, 2007,4 (5) :682-695.

[5] QIAN Yi, Hou Yi-bin. A fast string matching algorithm [J]. Mini-Micro Systems, 2004,25 (3),: 400 - 413.

[6] WANG Jian-guo, ZHENG Jia-Heng. BM string matching algorithm for an improved algorithm [J]. Computer Engineering and Science, 2007, 29 (5): 94-95.

[7] ZHANG Hong-mei, Fan Ming-yu. Pattern matching BM Algorithm [J]. Computer Applications .2009,26 (9) :3249-3252.

[8] YUAN Jing-bo, ZHENG Ji-sen, DING Shun-li. A kind of BM pattern matching algorithm [J] Computer Engineering and Applications, 2009,45 (17) :105-107.

[9] Wu Xi-hong, Ling Jie. BM pattern matching algorithm in [J]. Computer Engineering and Design, 2007,28 (1): 29-30.

[10] NAVARRO G,FREDRIKSSON K. Average complexity of exact and approximate multiple string matching[J].Theoretical Computer Sxience,2004, 321 (2-3) : 283-290.

[11] Knuth DE,Morris JH,Pratt VR.Fast pattern in strings[J]. SIAM Journal on Computing,1977,6(2):323-350.

[12] Lecroq T.Fast exact string matching algorithm[J]. Information Processing Letter.s. 2007,6(102):229-235.