Spam Filtering through Multiple Pattern Bit Parallel String Matching Combining Shift AND and OR

Vidya SaiKrishna
Department Of Computer Science And Engineering Maulana Azad National Institute Of Technology Bhopal-462051,

Akhtar Rasool
Asst. Professor Department Of Computer Science And Engineering Maulana Azad National Institute Of Technology Bhopal-462051,

Nilay Khare, PhD.
Associate Professor, HOD Department Of Computer Science And Engineering Maulana Azad National Institute Of Technology Bhopal-462051

ABSTRACT

Spam refers to unsolicited, unwanted and inappropriate bulk email. Spam filtering has become conspicuous as they consume a lot of network bandwidth, overloads the email server and drops the productivity of global economy. Content based spam filtering is accomplished with the help of multiple pattern string matching algorithm. Traditionally Aho Corasick algorithm was used to filter spam which constructs a trie of the spam keywords. The performance degrades in the context of time as well as space as the size of trie increases with the growing spam keywords count. To counterbalance time and space loss, bit parallel multiple pattern string matching algorithm using Shift OR method is used. The method acts as filter performing approximate string matching . This implies that there are some false matches detected by the filter which requires verification. The proposed method for filtering spams has been developed using a combination of Shift AND and OR operation. The method directly works on spam keywords of equal size whereas for unequal size keywords, a new proposed equal size grouping method is developed. Both method shows improvement over the Aho Corasick algorithm in context of space complexity and also behaves as an efficient filter and reducing the number of false matches as present in Shift OR method.

Keywords

Spam filtering, String matching, Bit parallelism .

1. INTRODUCTION

Spam refers to unsolicited, unwanted, inappropriate bulk email, Usenet postings and MUD/IRC monologs. The spam filter is used to disembody the spam mails from the collection of mails. The Spam filter designed in this paper is a content based filter that assess the words found in each individual message to find out whether an email is spam or legitimate.

The following sections describe types of spam filters, traditional method of spam filtering through Aho- Corasick algorithm and Bit parallel method of filtering spam. The Aho Corasick algorithm is a classic solution to string matching problem which builds a keyword tree called trie. The text string is matched against the AC Automaton having three kinds of actions. The Goto function gives the state entered from current state after matching the target character. The failure function gives the state returned after a mismatch. The output function returns the set of patterns recognized after entering a state. The filtering process becomes unrealistic when the pattern set enormously grows as the size of the trie increases.

Bit Parallelism using Shift OR is an approach to solve the string matching problem, where the pattern size is assumed to be less than or equal to word size of the computer. In this method , each character of the pattern is transformed into a bit vector. This transformation will result in faster searching because bit operations inside the system are faster and done in parallel. Working in this manner increases the speed of operation by a factor of almost w which is the word size of the computer.[5]

The filter developed using Shift OR method produces false matches which is mathematically calculated and compared with the proposed method. The false matches produced by it are enormous which leads to incorrect conclusion regarding spamacity of the email. This has led to the development of a better filter that produces less false matches and will more correctly categorize email as spam or ham. The proposed method uses a combination of Combined Shift OR and AND method to filter Spam which is compared with the Aho corasick algorithm and Shift OR method.

2. SPAM FILTERING

The spam filter is used to separate out the spam mails from the collection of mails.

2.1 Types of Spam Filters

Spam filters are mainly categorized as list based and content based spam filters.

List-Based Filters

List-based filters attempt to stop spam by categorizing senders as spammers or trusted users, and blocking or allowing their messages accordingly. The various types of filters in this category are Blacklist filters, Real time Blackhole list and Whitelist filters.[8]

Content Based Filters

Rather than enforcing across-the-board policies for all messages from a particular email or IP address, content-based filters evaluate words or phrases found in each individual message to determine whether an email is spam or legitimate.[8]

- **Heuristic Filters**

Rather than blocking messages that contain a suspicious word, heuristic filters take multiple terms found in an email into consideration. Heuristic filters scan the contents of incoming emails and assigning points to words or phrases. Suspicious words that are commonly found in spam messages, such as "Rolex" receive higher points, while terms frequently found in

normal emails receive lower scores. The filter then adds up all the points and calculates a total score. If the message receives a certain score or higher ,the filter identifies it as spam and blocks it. Messages that score lower than the target number are delivered to the user.[8]

- **Bayesian Filters**

Bayesian filters employ the laws of mathematical probability to determine which messages are legitimate and which are spam. In order for a Bayesian filter to effectively block spam, the end user must initially "train" it by manually flagging each message as either junk or legitimate. Over time, the filter takes words and phrases found in legitimate emails and adds them to a list; it does the same with terms found in spam. The type of filter developed in the paper is a Bayesian Filter.[8]

2.2 Bayesian Spam Filtering

It makes use of a naive Bayes classifier to identify spam e-mail. Bayesian classifiers work by correlating the use of tokens (typically words, or sometimes other things), with spam and non-spam e-mails and then using Bayesian inference to calculate a probability that an email is spam or not .

Computing the probability that a message containing a given word is spam

Suppose the suspected message contains the word "replica". The formula used is derived from Bayes theorem.[mention reference]

$\Pr(S|W) = \frac{\Pr(W|S)}{\Pr(W|S) + \Pr(W|H)}$ This quantity is called "spam city" (or "spaminess") of the word "replica". The number $\Pr(W|S)$ used is approximated to the frequency of messages containing "replica" in the messages identified as spam during the learning phase. Similarly, $\Pr(W|H)$ is approximated to the frequency of messages containing "replica" in the messages identified as ham during the learning phase. For these approximations to make sense, the set of learned messages needs to be big and representative enough. The Bayesian spam software tries to consider several words and combine their spamacities to determine a message's overall probability of being spam.

Combining individual probabilities

The individual probability is combined by using another formula is derived from Bayes' theorem:[8]

$$P = \frac{P1P2 \dots \dots . . PN}{P1P2 \dots . . PN + (1 - P1)(1 - P2) \dots \dots \dots . (1 - PN)}$$

where:

- P is the probability that the suspect message is spam;
- P1 is the probability $P(S|W1)$ that it is a spam knowing it contains a first word (for example "replica");
- P2 is the probability $P(S|W2)$ that it is a spam knowing it contains a second word (for example "watches");etc...
- PN is the probability $P(S|WN)$ that it is a spam knowing it contains an *N*th word (for example "home").

The result *p* is usually compared to a given threshold to decide whether the message is spam or not. If *p* is lower than the threshold, the message is considered as likely ham, otherwise it is considered as likely spam.[8]

3. SPAM FILTERING THROUGH SHIFT OR METHOD FOR EQUAL SIZED KEYWORDS

The words inside the email are indentified using the bit parallel pattern matching Rather than matching the text against exact patterns, the set of patterns is transformed to a single general pattern containing classes of characters. For example if there are three patterns, 'abcd' , 'pony', and 'abnh', the characters {a, p} are accepted in the first position, characters {b, o} in the second position, characters {c, n} in the third position and characters {d, h, y} in the fourth position.[1]

The NFA (Non-Deterministic Finite automaton) recognizing the set of character class pattern is built. Then the automaton is simulated using the bit parallel technique. This simulation results in faster execution, without having the need to construct the automaton.

The automaton has a transition from state (i-1) to i , if on reading the character "c" from the text , there is zero appearing in i^{th} position of vector D. Vector D is a state vector which is initialized to all 1's.[8] When the character c is read from the text, D is updated as D = (D<<1) | B[c] . If the 0 moves from least significant bit position to the most significant bit position of vector D, a match is reported. This situation corresponds to final state of the automaton, thereby recognizing the pattern. The pseudo code of the algorithm is given below.

The assumption in this method is that all the patterns $p_1p_2 \dots . p_r$ have equal size m and m<=w, where w is word size of the computer.[8]

ALGORITHM MultiplePatternSearch(text=t1t2.tn)

---

1. POS←0
2. textLength←stringLength(text)
3. N←0//N denotes Number Of Occurrence
4. $D=1^m$
5. While(POS<= textLength)
6. D=D<<1 | text[B[POS]]
7. If(MSB[D]=0)
8. POS←POS+1
9. N←N+1
10. GOTO Step 4
11. POS←POS+1
12. End of While

Example : Text= "hhello"

Pattern = { "hello", "world"}

The Automaton recognizing the set of patterns is shown in figure 1

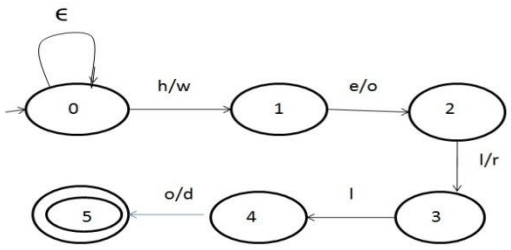

Figure 1 : NFA finding occurrence of character class pattern.

The character class pattern is "[h,w],[e,o],[l/r],[l],[o/d]"

The Bit Vectors are set in the following manner.

B[h]=11110, B[e]=11101,B[l]=10011,B[o]=01101 , B[w]=11110, B[r]=11011, B[d]=01111

Here 0 denotes occurrence and 1 denotes non occurrence of the character in different patterns.

Table 1 shows bit parallel simulation of above automata.

Table 1 : Multiple pattern search example

1	Text = hhello D 11110 B[h]11110 OR D 11110 D[1]=0 , so shift To next state	3.	Text = hhello D 11100 B[e] 11101 OR D 11101 D[2]=0, so shift to next state
2.	Text = hhello D 11100 B[h] 11110 OR D 11110 D[2]=1 , so it remains in the same state	4.	Text = hhello D 11010 B[l] 10011 OR D 11011 D[3]=0, so shift to next state
5.	Text = hhello D 10110 B[l]10011 OR D 10110 D[4]=0, so shift to next state	6.	Text = hhello D 01100 B[o] 01101 OR D 01101 D[5]=0, so shift to next State, which is the final state and the pattern is recognized.

4. PROPOSED MULTIPLE PATTERN MATCHING COMBINING SHIFT AND and OR OPERATIONS FOR EQUAL SIZED SPAM KEYWORD SET

The shift OR algorithm described above can be alternatively solved by the using the combination of shift AND and OR operation which results in less number of false matches. The bit vector is set in the following manner. Here bit 1 denotes occurrence and bit 0 denotes non occurrence of the characters in the pattern.

The automaton has a transition from state (i-1) to i , if on reading the character "c" from the text , there is a 1 appearing in $(m-i+1)^{th}$ position of vector D. Vector D is a state vector which is initialized to all 1's. The idea has been taken from the reference [7].

When the character c is read from the text, D is updated as $D = (D<<1 \mid 0^i1) \& B[c]$. If the 1 moves from most significant bit position to the least significant bit position of vector D, a match is reported. This situation corresponds to final state of the automaton, thereby recognizing the pattern. When a match is reported , vector D is again reinitialized to all ones and the process again proceeds in the same manner for further characters of the text. The pseudo code of the algorithm is given below.

ALGORITHM
CombinedShiftAndORPatternSearch(text=t1t2…..tn)

---

1. POS←0
2. textLength←stringLength(text)
3. N←0//N denotes Number Of Occurence
4. $D=0^m$;i=1
5. $orFactor=10^{m-1}$
6. While(POS<= textLength)
7. D=(D<<1 | orFactor) & text[B[POS]]
8. If (D & orFactor=0)
9. $orFactor=10^{m-1}$
10. POS←POS+1
11. GOTO step 4
12. If(LSB[D]=1)
13. POS←POS+1
14. N←N+1
15. GOTO Step 4
16. if ((D AND orFactor)= orFactor) and $i \leq m$)
17. $i\leftarrow i+1;orFactor=0^i10^{m-i-1}$
18. POS←POS+1
19. End of While

Example : Text= “hhello”

Pattern = { “hello”, “world”}

The bit vectors are set in the following manner

B[h]=10000, B[e]=01000, B[l]=00110, B[o]= 01001, B[w]=10000, B[r]=00100, B[d]=00001

The explanation is given in table 2

Table 2 : Combined Shift AND and OR pattern Matching

1	Text = hhello D 00000 OR 10000 B[h]10000 AND D 10000 D[5]=1 , so shift to next state	4.	Text = hhello D 10000 OR 00100 B[l] 00110 AND D 00100 D[3]=1 , so shift to next state
2.	Text = hhello D 00000 OR 01000 B[h]10000 AND D 00000 D[4]=0 , so it remains in the same state	5.	Text = hhello D 01000 OR 00010 B[l] 00110 AND D 00010 D[2]=1 , so shift to next state
3.	Text = hhello D 00000 OR 01000 B[e]01000 AND D 01000 D[4]=1 , so shift to next state	6.	Text = hhello D 00100 OR 00001 B[l] 01001 AND D 00001 D[1]=1 , we reach the final state Thereby reporting occurrence of the pattern

5. MATHEMATICAL ANALYSIS

Number of False Matches for Shift OR Method

We assume there is a pattern set P=(p1,p2……pk) of K patterns. All the patterns are assumed to be having equal length m. We are calculating the false matches for the worst case , where all the patterns are assumed to be having distinct characters in all pattern positions . In this case :

(i) Total Number of correct Matches(CM) = K, as recognized by the Automaton.

(ii) Total number of matches recognized by the automaton (TM)= K^m

(iii) Total Number of false matches(FM1) = Total Matches- Total number of Correct matches.

FM= $K^m – K$

(iv) In addition to these there are other false matches detected . Considering the following text and the pattern

Text : “heabcdello” and the pattern “hello” .

The Shift OR method will detect one pattern match in the above text. Counting the false matches for such case.

FM2= m(∑*-k) where ∑ denotes the size of the input alphabet .

(v) Total False Matches(FM)= FM1 + FM2

FM= $K^m – K + m(∑^*-k)$

= $K^m – K + m∑^* - mk$

= $K\{ K^{m-1} –m-1\} + m∑^*$

Number of False Matches for combined Shift AND and OR Method

The filtering efficiency has been improved in the combined method. This is mainly because of the fact that in the present state if the character is not accepted then the combined method forces to leave the present state and again restarts from the initial state. This is understood from the figure 2.

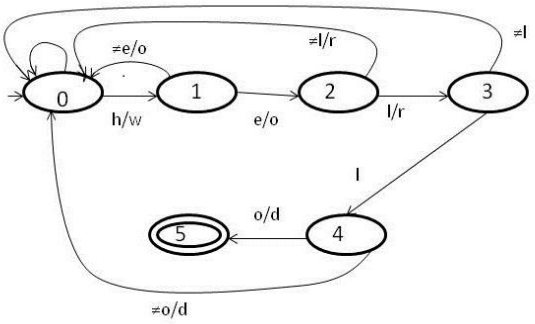

Figure 2: NFA recognizing character class pattern

Text : “hefffffffxfdllo” and the pattern “hello” .

The Shift OR method will detect one pattern match in the above text. If the same pattern is found using the combined Shift AND and OR method, no pattern match will be detected.

(i) The Number of false matches is only equal to

(FM1) = Total Matches - Total number of Correct matches i.e FM= $K^m – K= K(K^{m-1} -1)$

(ii) Besides this there are no false matches detected by the automaton, which shows an improvement over the Shift OR method.

Assumption of above methods

Both the methods assume that the spam keywords are all of equal sizes. To make the filtering process more realistic , another proposed method has been developed which considers keywords of unequal size.

6. PROPOSED GROUPED METHOD FOR UNEQUAL SIZED SPAM KEYWORDS

The proposed method removes demerit of the above method i.e the method works for patterns of unequal size. This is done by grouping patterns of equal size and then the processing takes place individually for each group. The algorithm is named as Grouped-Multiple Pattern algorithm. The pseudo code for the method is given below.

ALGORITHM Grouped_Multiple_Pattern(text=t1t2…..tn)

---

1. textLength←stringLength(text)
2. N←0// N denotes Number Of Occurrence
3. For i=1 to noOfGroups do
4. Begin
5. POS←0
6. set the bit vector for group[i]
7. // group[i] is available in the form linked list
8. $D=1^m$
9. While(POS<= textLength)
10. D=D<<1 | text[B[POS]]
11. If(MSB[D]=0)
12. POS←POS+1
13. N←N+1
14. GOTO Step 8
15. POS←POS+1
16. End of While
17. End

Example :

Patterns = (lift, time , hello , world , display)

Group 1 : {lift, time}

Group 2 : {hello , world}

Group 3 : { display}

Bit Vectors :

group 1 :B[l]=1110, B[i]=1101, B[f]=1011,
B[t]=0110, B[m]=1011 , B[e]=0111

group 2 : B[h]=11110, B[e]=11101, B[l]=10011,
B[o]=01101 , B[w]=11110, B[r]=11011,
B[d]=01111

group 3 : B[d]=1111110 , B[i]=1111101, B[s]=
1111011, B[p]=1110111, B[l]=1101111,
B[a]= 1011111, B[y]=0111111

The bit vectors are individually set for each group and the same code gets executed for each group. Working in this manner makes the process of spam filtering more realistic, as this works for patterns of unequal size.

7. EXPERIMENTAL RESULTS

Experiment was conducted using Enron Spam dataset which is available from the http://www.iit.demokritos.gr/skel/i-config/ and http://www.aueb.gr/users/ion/publications.html in both raw and pre-processed form. The "preprocessed" subdirectory contains the messages in the preprocessed format that is used in the experiments of the paper. Each message is its separate text file.In particular Enron1 Spam data is used which has a collection of 5975 emails , which includes 1500 spam emails. Few spam emails of variable size were taken into consideration, which filtered using the conventional Shift OR method and the proposed method .

Both the methods were compared on the basis of number of matches and the spamacity factor. The following graph in figure 3 and table 3 compares the performance on the basis of Number of Counts.

Table 3 : Comparison on the basis of Number of Matches

Text Size In KB	Number Of Matches	
	Shift OR	Combined Shift AND and OR
1	38	29
1	169	121
2	65	44
3	220	165
3	115	93
9	179	162

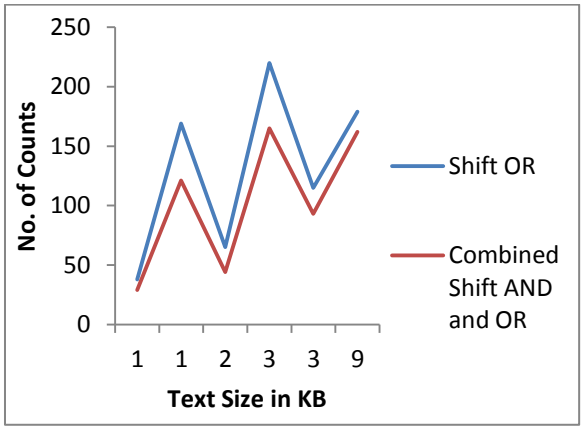

Figure 3: Comparing Number of Counts

As clear from the above graph that the Combined Shift AND and OR method results in less number of false matches than the shift OR method. The performance is also compared on the basis of spamacity as shown in figure 4 . The filter uses the Bayes theorm to calculate the probability of occurences of the keywords stored in the database. Depending upon the probabilty calculated , a net score is computed . The net score determines the spamacity of the email. If the net score is above the threshold, the email is classified as spam otherwise

ham.Large number of false matches may result in more spamacity of the probable spam keywords and this might lead to incorrectly categorize the email , if filtered through Shift OR method.

Table 4 Comparing Spamacity

Text Size In KB	Spamacity	
	Shift OR	Combined Shift AND and OR
1	9.98	9.5
1	9.9999	9.189
2	9.09	8.876
3	9.95215	9.5877
3	7.9673	7.6765
9	9.9	9.3

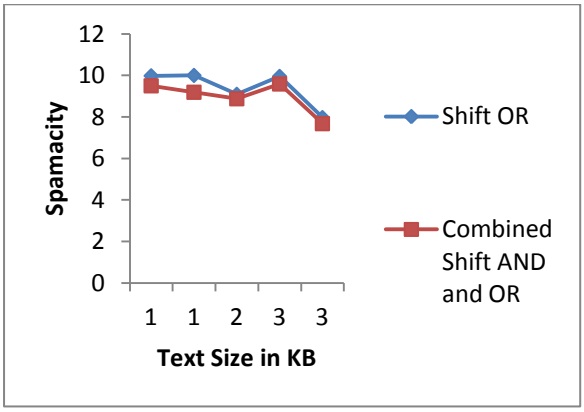

Figure 4: Comparing Spamacity

8. CONCLUSION

Bit Parallel multiple pattern string matching algorithm using combined Shift AND and OR operation is used as a content based spam filter to classify emails as spam or ham. The advantage of using bit parallel approach is that the process of matching the keywords against the email is faster. The method also results in less number of false matches , thereby more correctly classifying the emails as spam or ham.

9. FUTURE WORK

In future ,we are trying to extend the work by having a parallel version of the proposed algorithm , so that it can be on any of the parallel architecture. Unit).

10. REFERENCES

[1] Leena Salmela, J. Tarhio and J. Kytojoki "MultiPattern String Matching with Very Large Pattern Sets", ACM Journal of Experimental Algorithmics, Volume 11, 2006. Rajesh Prasad, Anuj Kumar Sharma and Alok Singh.

[2] Rajesh Prasad, Anuj Kumar Sharma and Alok Singh. "Efficient bit parallel multiple Pattern approximate string matching algorithm", Academic Journals , Scientific Research and Essays, Vol(6), pp876-881, February 2011.

[3] Gonzalo Navarro and Mathieu Raffinot. A Bit Parallel approach to Suffix Automata : Fast Extended String Matching. In M. Farach (editor), Proc. CPM'98, LNCS 1448. Pages 14-33, 1998.

[4] G. Navarro,M. Raffinot, Fast and flexible string matching by combining bit-parallelism and suffix automata,ACM J. Experimental Algorithmics (JEA) 5 (4) (2000).

[5] David E. Culler, Jaswinder Pal Singh, Anoop Gupta. Parallel Computer Architecture - A Hardware/Software Approach. Morgan Kaufmann Publishers, 1999. ISBN 1-55860-343-3, pg 15

[6] "The Two Percent Solution" by Jim Turley 2002 [6] http://en.wikipedia.org/wiki/string_searching_algorthim

[7] Baeza-Yates RA, Gonnet GH (1992). A new approach to text searching.Commun. ACM., 35(10): 74-82.

[8] Dylan Mors and Dermot Harnett (2009)

[9] Paul Graham (2002) A Plan for Spam