Global Journal of Computer Science and Technology
Volume 12 Issue 1 Version 1.0 January 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

An Enhanced Cuckoo Search for Optimization of Bloom Filter in Spam Filtering

By Arulanand Natarajan, Subramanian S, Premalatha K
Anna University of Technology, Coimbatore

Abstract - Bloom Filter (BF) is a simple but powerful data structure that can check membership to a static set. The trade-off to use Bloom filter is a certain configurable risk of false positives. The odds of a false positive can be made very low if the hash bitmap is sufficiently large. Spam is an irrelevant or inappropriate message sent on the internet to a large number of newsgroups or users. A spam word is a list of well-known words that often appear in spam mails. The proposed system of Bin Bloom Filter (BBF) groups the words into number of bins with different false positive rates based on the weights of the spam words. An Enhanced Cuckoo Search (ECS) algorithm is employed to minimize the total membership invalidation cost of the BFs by finding the optimal false positive rates and number of elements stored in every bin. The experimental results have demonstrated for CS and ECS for various numbers of bins.

Keywords : *Bin Bloom Filter, Bloom Filter, Cuckoo Search, Enhanced Cuckoo Search, False positive rate, Hash function, Spam word.*

GJCST Classification: *D.3.4 , G.1.6, B.1.4*

Strictly as per the compliance and regulations of:

An Enhanced Cuckoo Search for Optimization of Bloom Filter in Spam Filtering

Arulanand Natarajan [α], Subramanian S [Ω], Premalatha K [β]

Abstract - Bloom Filter (BF) is a simple but powerful data structure that can check membership to a static set. The trade-off to use Bloom filter is a certain configurable risk of false positives. The odds of a false positive can be made very low if the hash bitmap is sufficiently large. Spam is an irrelevant or inappropriate message sent on the internet to a large number of newsgroups or users. A spam word is a list of well-known words that often appear in spam mails. The proposed system of Bin Bloom Filter (BBF) groups the words into number of bins with different false positive rates based on the weights of the spam words. An Enhanced Cuckoo Search (ECS) algorithm is employed to minimize the total membership invalidation cost of the BFs by finding the optimal false positive rates and number of elements stored in every bin. The experimental results have demonstrated for CS and ECS for various numbers of bins.

Keywords : Bin Bloom Filter, Bloom Filter, Cuckoo Search, Enhanced Cuckoo Search, False positive rate, Hash function, Spam word.

I. Introduction

A spam filter is a program that is used to detect unsolicited and unwanted email and prevent those messages from getting into user's inbox. A spam filter looks for certain criteria on which it stands decisions. For example, it can be set to look for particular words in the subject line of messages and to exclude these from the user's inbox. This method is not effective, because often it is omitting perfectly legitimate messages and letting actual spam through. The strategies used to block spam are diverse and includes many promising techniques. Some of the strategies like black list filter, white list /verification filters rule based ranking and naïve bayesian filtering are used to identify the spam.

A BF presents a very attractive option for string matching (Bloom 1970). It is a space efficient randomized data structure that stores a set of signatures efficiently by computing multiple hash functions on each member of the set.

It queries a database of strings to verify for the membership of a particular string. The answer to this query can be a false positive but never be a false negative. The computation time required for performing the query is independent of the number of signatures in the database and the amount of memory required by a BF for each signature is independent of its length (Feng et al 2002).

This paper presents a BBF which allocates different false positive rates to different strings depending on the significance of spam words and gives a solution to make the total membership invalidation cost minimum. BBF groups strings into different bins via smoothing by bin means technique. The number of strings to be grouped and false positive rate of each bin is identified through GA which minimizes the total membership invalidation cost. This paper examines different number of bins for given set of strings, their false positive rates and number of strings in every bin to minimize the total membership invalidation cost.

The organization of this paper is as follows. Section 2 deals with the standard BF. Section 3 presents the CS technique. Section 4 reports optimized BBF using ECS. Performance evaluation of CS and ECS for the BBF is discussed in section 5.1

II. Bloom Filter

Bloom filters (Bloom 1970) are compact data structures for probabilistic representation of a set in order to support membership queries. This compact representation is the payoff for allowing a small rate of false positives in membership queries which might incorrectly recognize an element as member of the set.

Given a string S the BF computes k hash functions on it producing k hash values and sets k bits in an m-bit long vector at the addresses corresponding to the k hash values. The value of k ranges from 1 to m. The same procedure is repeated for all the members of the set. This process is called programming of the filter. The query process is similar to programming, where a string whose membership is to be verified is input to the filter. The bits in the m-bit long vector at the locations corresponding to the k hash values are looked up. If at least one of these k bits is not found in the set then the string is declared to be a nonmember of the set. If all the bits are found to be set then the string is said to belong to the set with a certain probability. This uncertainty in the membership comes from the fact that those k bits in the m-bit vector can be set by any other n-1 members. Thus finding a bit set does not necessarily imply that it was set by the particular string being queried. However, finding a bit not set certainly implies that the string does

Author α : Anna University of Technology, Coimbatore.
E-mail : arulnat@yahoo.com
Author Ω : Sri Krishna College of Engineering and Technology, Coimbatore. E-mail : dsraju49@gmail.com
Author β : Bannari Amman Institute of Technology, Erode.
E-mail : kpl_barath@yahoo.co.in

not belong to the set. In order to store a given element into the bit array, each hash function must be applied to it and, based on the return value r of each function (r1, r2, … , rk), the bit with the offset r is set to 1. Since there are k hash functions, up to k bits in the bit array are set to 1 (it might be less because several hash functions might return the same value). Figure 1 is an example where m=16, k=4 and e is the element to be stored in the bit array.

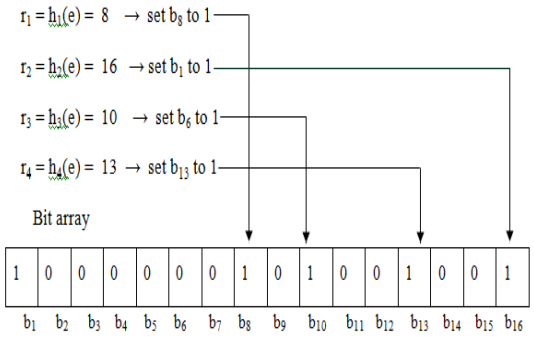


Fig.1 : Bloom Filter

One important feature of BF is that there is a clear tradeoff between the size of the filter and the rate of false positives. The false positive rate of BF is

$$f = \left(1 - e^{-kn/m}\right)^k = \exp(k \ln(1 - e^{-kn/m})) \quad (1)$$

Let $g = k \ln(1 - e^{-kn/m})$. Minimizing the false positive probability f is equivalent to minimizing with respect to k.

$$\frac{dg}{dk} = \ln\left(1 - e^{-\frac{kn}{m}}\right) + \frac{kn}{m} \frac{e^{-kn/m}}{1 - e^{-kn/m}} \quad (2)$$

The derivative equals 0 when $k_{min} = (\ln 2)(m/n)$. In this case the false positive probability f is:

$$f(k_{min}) = (1 - p)^{k_{min}} = \left(\frac{1}{2}\right)^{k_{min}} = (0.6185)^{m/n} \quad (3)$$

of course k should be an integer, so k is $\lceil \ln 2 \cdot (m/n) \rceil$

The BF has been widely used in many database applications (Mullin 1990; Mackert and Lohman, 1986). It is applied in networking literature (Brooder and Mitzenmacher, 2005). A BF can be used as a summarizing technique to aid global collaboration in peer-to-peer networks (Kubiatowicz et al., 2000 ; Li et al, 2002 ; Cuena-Acuna et al, 2003). It supports probabilistic algorithms for routing and locating resources (Rhea and Kubiatowicz 2004; Hodes et al,2002 ; Reynolds and Vahdat, 2003; Bauer et al, 2004) and share Web cache information (Fan et al,2000). BFs have great potential for representing a set in main memory (Peter and Panagiotis, 2004) in stand-alone applications. BFs have been used to provide a probabilistic approach for explicit state model checking of finite-state transition systems (Peter and Panagiotis, 2004). It is used to summarize the contents of stream data in memory (Jin et al,2004; Deng and Rafiei,2006), to store the states of flows in the on-chip memory at networking devices (Bonomi et al,2006), and to store the statistical values of tokens to speed up the statistical-based Bayesian filters (Li and Zhong,2006). The variations of BFs are compressed Bloom filters (Mitzenmacher,2002),counting Bloom filters (Fan et al,2000), distance-sensitive Bloom filters (Kirsch and Mitzenmacher,2006), Bloom filters with two hash functions (Kirsch and Mitzenmacher,2006), spacecode Bloom filters (Kumar et al,2004), spectral Bloom filters (Cohen and Matias,2003), generalized Bloom filters (Laufer et al,2005), Bloomier filters (Chazelle et al,2004), and Bloom filters based on partitioned hashing (Hao et al,2007).

III. Cuckoo Search

Cuckoo search is an optimization algorithm inspired by the brood parasitism of cuckoo species by laying their eggs in the nests of other host birds proposed by Yang and Deb (2009). If a host bird discovers the eggs are not their own, it will either throw these foreign eggs away or simply abandon its nest and build a new nest elsewhere. Each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The better new solution (cuckoo) is replaced with a solution which is not so good in the nest. In the simplest form, each nest has one egg. When generating a new solution Levy flight is performed. The rules for CS are described as follows:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest
- The best nests with high quality of eggs will carry over to the next generations;
- The number of available host nests is fixed, and a host can discover an foreign egg with a probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location

The algorithm for CS is given below:

Generate an initial population of n host nests;

while (t<MaxGeneration) or (stop criterion)

Get a cuckoo randomly (say, i) and replace its solution by performing Levy flights;

Evaluate its fitness Fi

Choose a nest among n (say, j) randomly;

if (Fi > Fj),[for maximization]

Replace j by the new solution;

end if

A fraction (pa) of the worse nests is abandoned and new ones are built;

Keep the best solutions/nests;

Rank the solutions/nests and find the current best;

Pass the current best to the next generation;

end while

IV. Enhanced Cuckoo Search for Bloom Filter Optimization

a) Bin Bloom Filter (BBF)

A BBF is a date structure considering weight for spam word. It groups spam words into different bins depending on their weight. It incorporates the information on the spam word weights and the membership likelihood of the spam words into its optimal design. In BBF a high cost bin lower false positive probability and a low cost bin has higher false positive probability. The false positive rate and number of strings to be stored is identified through optimization technique GA which minimize the total membership invalidation cost. Figure 2 shows Bin BF with its tuple <n,f,w> configuration.

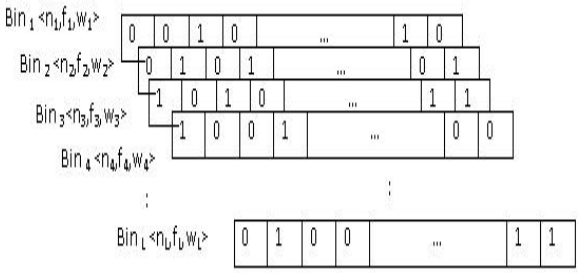


Fig.2 : Bin Bloom Filter

b) Problem Definition

Consider a standard supervised learning problem with a set of training data D = {<Y1,Z1 >,..., <Yi, Zi>, … ,< Yr ,Zr >} , where Yi is an instance represented as a single feature vector, Zi = C(Yi) is the target value of Yi , where C is the target function. Where Y1, Y2, … , Yr set of text document collection C is a class label to classify into spam or legitimate (non-spam). The collection is represented into feature vector by the text documents are converted to normalized case, and tokenized them, splitting on non-letters. The stop words are eliminated. The spam weights for words are calculated from the set. This weight value indicates its probable belongings to spam or legitimate. The weight values are discretized and assigned for different Bins. The tuple to describe the Bin BF is, {{n1, n2,, …, nL}, {w1, w2,…, wL}, m, {k1, k2, …, kL}, {f1, f2, …, fL}}. It is an optimization problem to find the value of n and f that to minimize the total membership invalidation cost. For membership testing the total cost of the set is the sum of the invalidation cost of each subset. The total membership invalidation cost (Xie et al., 2005) is given as,

$$F= n1f1w1 + n2f2w2 + \ldots\ldots + nLfLwL$$

The total membership invalidation cost

$$F(L) = \sum_{i=1}^{L} n_i w_i f_i \qquad (4)$$

to be minimized.

Where $$\sum_{i=1}^{L} n_i = N$$

N- Total number of Strings in a spam set.

$$f_i = \left(\frac{1}{2}\right)^{\ln 2 \times \left(r_i m \Big/ \sum_{j=1}^{i} n_j r_j \right)}$$

$$r_i = \ln(f_i) \quad (1 \le i \le L)$$

The objective function f(L) taken as standard for the problem of minimization is

$$f(L) = \begin{cases} C_{max} - F(L) & \text{if } F(L) < C_{max} \\ 0 & \text{if } F(L) \ge C_{max} \end{cases} \qquad (5)$$

where Cmax is a large constant.

c) ECS for Optimization of BF

The CS is extended to an ECS in which each nest has multiple eggs representing a set of solutions. Generate an initial population of n host nests with m eggs;

while (t<MaxGeneration) or (stop criterion)

Get a cuckoo randomly (say, i) by Levy flights using the best egg in the chosen nest;

Evaluate its fitness Fi

Choose a nest among n and choose an egg with the worst solution in the nest (say, j);

if (Fi > Fj),[for maximization]

Replace j by the new solution i;

end if

Find the best solution (among m) in each nest;

Rank the nests based on the best solution;

Abandon a fraction (pa) of the nests which have worse solutions and built new ones;

Keep the best solutions/nests;

Rank the solutions/nests and find the current best;

end while

When generating new solutions x(t+1) for a cuckoo i, a Levy flight is performed using the following equation (6)

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Levy}(\lambda) \quad (6)$$

The symbol ⊕ is an entry-wise multiplication. Basically Levy flights provide a random walk while their random steps are drawn from a Levy distribution for large steps

$$\text{Levy} \sim u = t^{-\lambda} \quad (7)$$

which has an infinite variance with an infinite mean. Here the consecutive jumps of a cuckoo essentially form a random walk process which obeys a power-law step- length distribution with a heavy tail. The representation of egg (solution) is given in figure 3.

$x_i =$ | n_{i1} | f_{i1} | w_{i1} | n_{i2} | f_{i2} | w_{i2} | … | n_{ij} | f_{ij} | w_{ij} | … | n_{iL} | f_{iL} | w_{iL} |

Fig.3 : Egg representations for Bin Bloom Filter

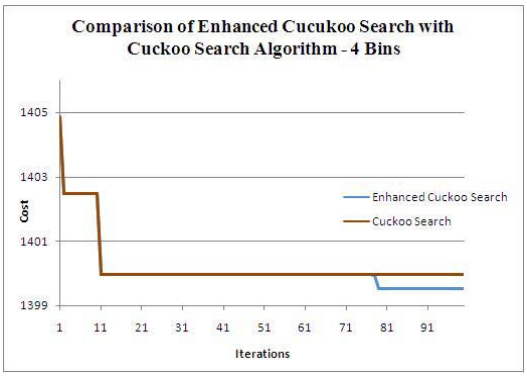
where nij, fij and wij refer respectively the number of words, false positive rate of and the weight of the jth bin of ith egg. The triplet <n,f,w> encodes a single bin. The false positive rate fij can be obtained from equation (1) where nij is drawn from the ith egg in the nest, m is known in advance and k is calculated from equation (3). One egg in the nest represents one possible solution for assigning the triples <n, f, w>. At the initial stage, each egg randomly chooses different <n, f, w> for L Bins based on the given constraints. The fitness function for each egg can be calculated based on the equation (5).

VII. Experimental Results

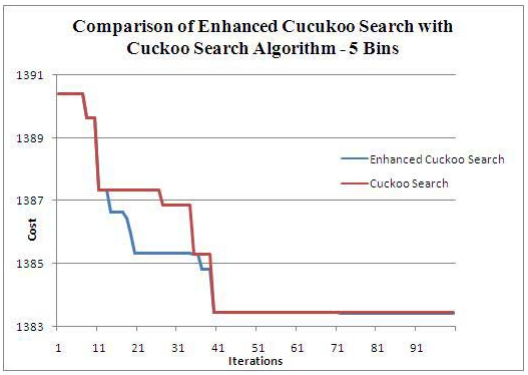
Cuckoo Search employs Levy flight for finding new solutions from equation (7). CS and ECS consider 10 nests and 50 iterations. The parameters pa,α and λ are set as 0.3, 1 and 1.5 respectively. The total number of strings taken for testing is 250, 500, and 1000. The string weights are varying from 0.0005 to 5. The size of the BF is 1024. This experimental setup is applied for number of bins from 4 to 7.

Figures 4a, 4b, 4c and 4d correspondingly show the total membership invalidation cost obtained from BBF for bin sizes from 4 to 7 for 1000 strings using CS and ECS algorithm. In this experimental setup the ECS performs better than CS. Figures 5a, 5b, 5c and 5d show the total membership invalidation cost obtained from BBF for bin sizes from 4 to 7 respectively for 500 strings. Figures 6a, 6b, 6c and 6d show the cost of BBF from bin sizes 4 to 7 for 250 strings. For all the string sizes the ECS outperforms CS.

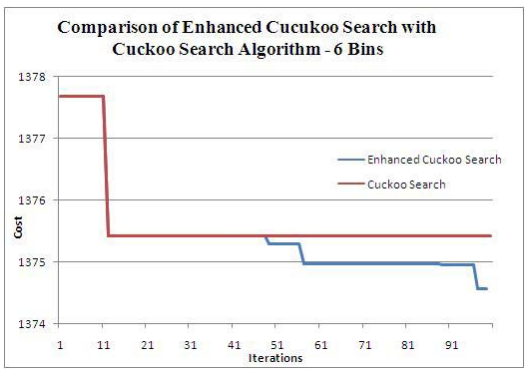
In CS, 10 nests which equals to number of nests in ECS and 40 nests which equals to number of eggs in ECS are taken to find the total membership invalidation cost for 1000 strings. Figure 7 shows the total membership invalidation cost obtained from BBF for the bin sizes ranging from 4 to 10 using CS and ECS. It shows that the cost is decreased when the numbers of bins are increased. The results obtained from ECS outperform CS for all bin sizes from 4 to 10.



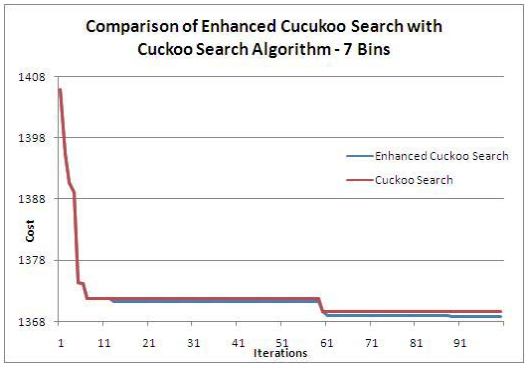
(A)



(B)

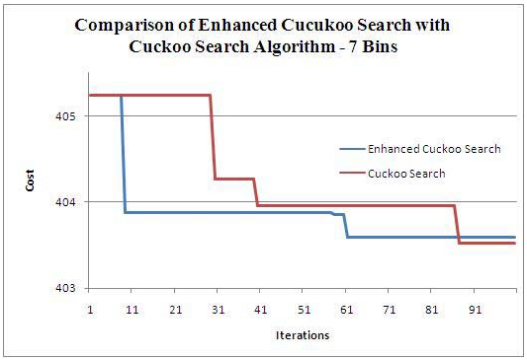


(C)



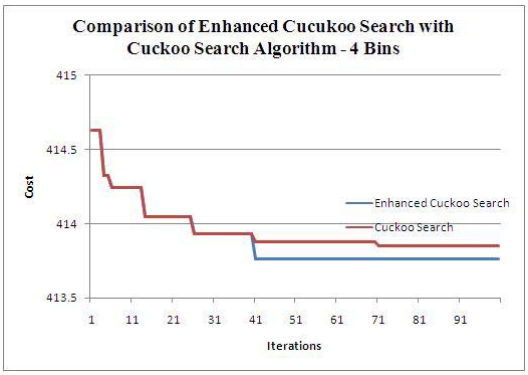
(D)

Fig.4 : Values obtained for 1000 Strings

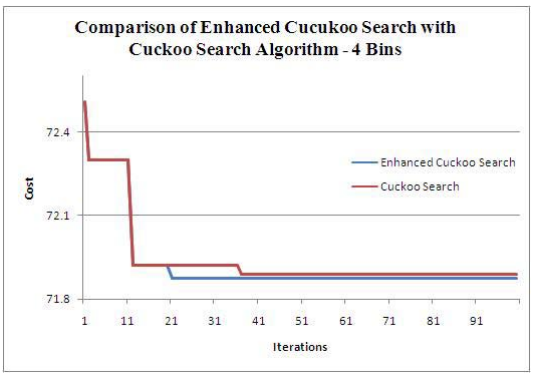


(D)

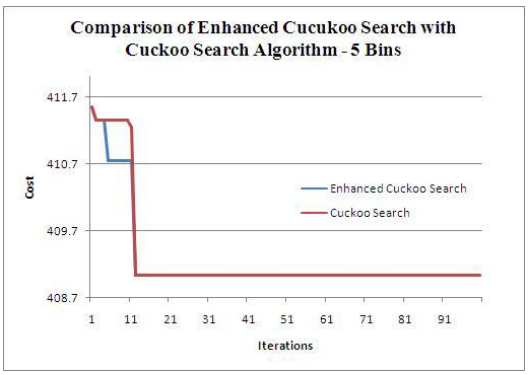
Fig.5 : Values obtained for 500 Strings



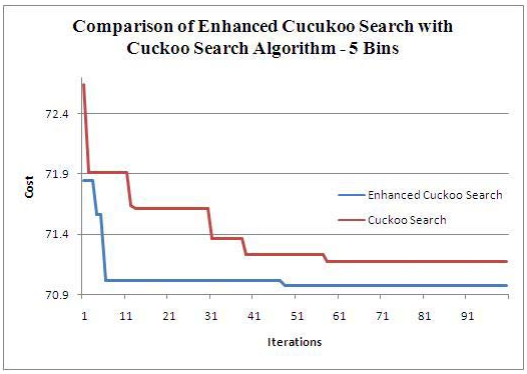
(A)



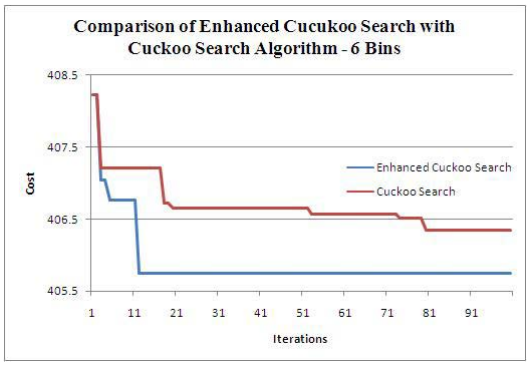
(A)



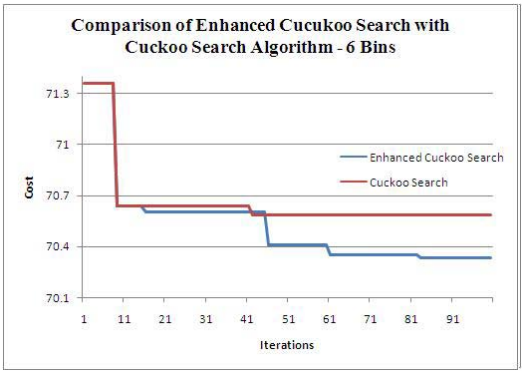
(B)



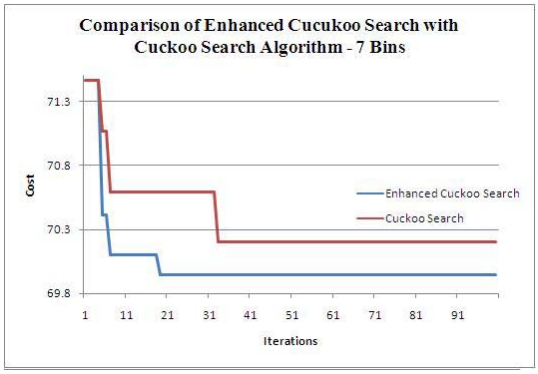
(B)



(C)



(C)



(D)

Fig.6 : Values obtained for 250 Strings

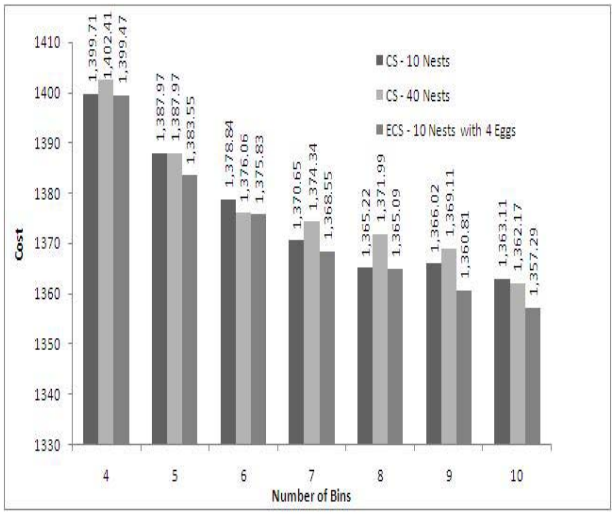


Fig.7 : Total Membership invalidation cost for CS and ECS

VIII. Conclusion

BFs are simple randomized data structures that are useful in practice. The BBF is an extension of BF, and inherits the best feature of BF such as time and space saving. The BBF treats strings in a set in a different way depending on their significance, groups the strings into bins and allocates different false positive rate to different bins. Important spam words have lower false positive rate than less significant words. In this work, we have applied CS and ECS for optimization of BF. The proposed system ECS outperforms CS.

References Références Referencias

1. Bloom B, Space/time tradeoffs in hash coding with allowable errors, Communications of the ACM, 13, 1970, 422–426.
2. Feng W,.Shin K.G, Kandlur D.D. & D.Saha, "The BLUE active queue management algorithms", IEEE/ACM Transactions on Networking, 10, 2002, 513 – 528.
3. Mullin J.K, Optimal Semijoins for Distributed Database Systems, IEEE Trans. Software Eng., 16, 1990, 558-560.
4. Mackert L.F. and Lohman G.M., Optimizer Validation and Performance Evaluation for Distributed Queries, Proc. 12th Int'l Conf. Very Large Data Bases (VLDB), 1986, 149-159.
5. Broder A and Mitzenmacher M. Network Applications of Bloom Filters: A Survey, Internet Math., 1(4), 2005, 485-509.
6. Kubiatowicz J Bindel D, Chen, Y Czerwinski S, Eaton P, and Geels D, Oceanstore: An Architecture for Global-Scale Persistent Storage," ACM SIGPLAN Notices, 35(11), 2000, 190-201.
7. Li J, Taylor J, Serban L, and Seltzer M, Self-Organization in Peer-to-Peer System, Proc. ACM SIGOPS, 2002.
8. Cuena-Acuna F.M, Peery C,Martin R.P, and Nguyen T.D, PlantP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities, Proc. 12th IEEE Int'lSymp. High Performance Distributed Computing, 2003, 236-249
9. Rhea S.C and Kubiatowicz J, Probabilistic Location and Routing, Proc. IEEE INFOCOM, 2004, 1248-1257.
10. Hodes T.D, Czerwinski S.E, and Zhao B.Y, An Architecture for Secure Wide Area Service Discovery, Wireless Networks, vol. 8, nos. 2/3, 2002, 213-230.
11. Reynolds P and Vahdat A, Efficient Peer-to-Peer Keyword Searching, Proc. ACM Int'l Middleware Conf., 2003, 21-40.
12. Bauer D, Hurley P, Pletka R, and Waldvogel M, Bringing Efficient Advanced Queries to Distributed Hash Tables, Proc. IEEE Conf. Local Computer Networks, 2004, 6-14
13. Fan L, Cao P, Almeida J, and Broder A, Summary Cache: A Scalable Wide Area Web Cache Sharing Protocol, IEEE/ACM Trans. Networking, 8(3), 2000, 281-293.
14. Peter C.D and Panagiotis M, Bloom Filters in Probabilistic Verification, Proc. Fifth Int'l Conf. Formal Methods in Computer- Aided Design, 2004, 367-381.
15. Jin C, Qian W, and Zhou A, Analysis and Management of Streaming Data: A Survey, J. Software, 15(8), 2004, 1172-1181.
16. Deng F and Rafiei D, "Approximately Detecting Duplicates for Streaming Data Using Stable Bloom Filters," Proc. 25th ACMSIGMOD, 2006, 25-36.
17. Bonomi F, Mitzenmacher M, Panigrahy R, Singh S, andVarghese G, Beyond Bloom Filters: From Approximate Membership Checks to Approximate State Machines, Proc. ACM SIGCOMM, 2006 , 315-326.
18. Li K and Zhong Z, Fast Statistical Spam Filter by

Approximate Classifications, Proc. Joint Int'l Conf. Measurement and Modeling of Computer Systems, SIGMETRICS/Performance, 2006, 347-358.
19. Mitzenmacher M, Compressed Bloom Filters, IEEE/ACM Trans.Networking, 10(5) 2002, 604-612.
20. Kirsch A and Mitzenmacher M, Building a Better Bloom Filter, Technical Report tr-02-05.pdf, Dept. of Computer Science, Harvard Univ,2006.
21. Kirsch A and Mitzenmacher M, Distance-Sensitive Bloom Filters, Proc. Eighth Workshop Algorithm Eng. and Experiments (ALENEX '06), 2006.
22. Kumar A, Xu J, Wang J, Spatschek O, and Li L, Space-Code Bloom Filter for Efficient Per-Flow Traffic Measurement, Proc. 23rd IEEE INFOCOM, 2004, 1762-1773.
23. Cohen S and Matias Y, Spectral Bloom Filters, Proc. 22nd ACM SIGMOD, 2003, 241-252.
24. Laufer R.P, Velloso P.B, and Duarte O.C.M.B, GeneralizedBloom Filters, Technical Report Research Report GTA-05-43, Univ. of California, Los Angeles (UCLA), 2005.
25. Chazelle B, Kilian J, Rubinfeld R, and Tal A, The Bloomier Filter: An Efficient Data Structure for Static Support Lookup Tables, Proc. Fifth Ann. ACM-SIAM Symp. Discrete Algorithms (SODA), 2004, 30-39.
26. Hao F, Kodialam M, and Lakshman T.V, Building High Accuracy Bloom Filters Using Partitioned Hashing, Proc. SIGMETRICS/Performance, 2007, 277-287.
27. Yang X.S., Deb S. "Cuckoo search via Lévy flights". World Congress on Nature & Biologically Inspired Computing (NaBIC 2009). IEEE Publications. 2009, 210–214.
28. Xie K., Min Y., Zhang D., Wen J., Xie G. & Wen J, Basket Bloom Filters for Membership Queries, Proceedings of IEEE Tencon'05,2005, 1-6.

Global Journals Inc. (US) Guidelines Handbook 2012

www.GlobalJournals.org