This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2009 proceedings.

Analyzing the Optimal Use of Bloom Filters in Wireless Sensor Networks Storing Replicas

Christine Jardak, Janne Riihijärvi and Petri Mähönen
Department of Wireless Networks, RWTH Aachen University
Kackertstrasse 9, D-52072 Aachen, Germany
Email: {cja,jar,pma}@mobnets.rwth-aachen.de

***Abstract*—We study the problem of optimal parameter selection for Bloom filters in content-based routing. In constrained environments, such as sensor network, optimizing lengths of the filters used and the number of hash functions employed can lead to significant reduction of memory overhead and false positives. We demonstrate by analytical calculations and extensive simulations that the commonly used heuristics for choosing these parameters are suboptimal especially in networks storing replicas of the data for redundancy. We show specifically how network topology and use of replication can be taken into account when selecting the parameter values used. Our results show that memory consumption can be reduced by up to 50% compared to usual approaches, and that the occurrence of false positives can be significantly reduced as well.**

I. INTRODUCTION

Content-based routing (CBR) using Bloom filters (BFs) [1] is a widely used method for making forwarding decisions based on packet payload or a query for particular data without explicit use of node addresses. The classical approach is to map a block of data into a bit sequence called *key* using a set of hash functions. This mapping is done by using the outputs of the hash functions to indicate locations of bits to be set to ones, the rest of the bits being zero. Each node then maintains a routing table consisting of a matrix of entries, where each row maps a hop count and each column refers to a direct neighbor. Each entry is a Bloom filter holding keys of the blocks data located certain hops away. The Bloom filter then consists of the logical OR of these keys. Checking whether a particular data block is a member of the Bloom filter, the key of that data block is compared to the value of the Bloom filter. If the corresponding bits set to one in the key are also set to one in the filter, membership is assumed with a certain probability of false positive. This mechanism suits very well for low power devices such as TelosB motes which have also limited RAM resources [2].

In order to increase survivability as well as to reduce the retrieval overhead, we treat the case when stored data is replicated in randomly selected nodes. In such networks using replicas, we highlight two common approaches used for choosing the number of hash functions in BFs and detail their drawbacks. The first common approach is to calculate the number of hash functions k minimizing the probability of error in a Bloom filter taking into account the total number of elements the filter. We conduct a general analytical calculation proving that in a network storing replicas the number of distinct keys in a filter is less than the total number of keys. The second approach is to calculate the number of k used in a network by considering the maximum number of keys inserted in an entry (Bloom filter) of the routing table. Practically, this means that the value of k is calculated taking into account the number of keys that are inserted in the last entries of the routing table. Our analysis proves that the value of k minimizing the probability of error in an entry of row i does not necessarily minimize it for the whole neighborhood consisting of all the entries from 1 until i. Our simulation results for both proposed solutions show the use of the optimal value of k minimizing at maximum the probability of false positives. This results into a higher probability of data localization. Moreover, we present a way of benefiting from the presence of replicas in the network in order to shorten the length of the BFs in use, decreasing by that its need for memory space.

The remainder of this paper is organized as follows. In Section II we briefly describe the common idea behind using BFs in content-based routing. Section III is the main section where we describe the possible problems of two major approaches that were traditionally used when applying Bloom filters for content-based routing. Moreover, this section provides a method for optimizing the length of the BFs in use. We present proposals in Section IV leading to an optimal probability of data localization and minimal memory usage. Section V describes the simulation. In Section VI we develop a simulation model allowing us to validate the proposed solutions for any complex network and show results for a tree and grid network topologies. We conclude the paper in Section VII.

II. CONTENT-BASED ROUTING AND BLOOM FILTERS

In the introduction we explained the concept of CBR using BFs we shall now detail the steps that take place when localizing a block of data and show the possible routing error. Once a new block of data has been stored, the storing node generates a key which identifies this block [3]. The key is then flooded to all nodes that are located less than a prescribed number of hops away. The nodes store the received key in their routing table consisting of a matrix of entries, with one column for each neighbor and with rows signifying the distance in number of hops to the node data is stored at. In order to keep the routing table compact, the receiving node executes a logical OR operation for the new keys with the already stored

978-1-4244-2948-6/09/$25.00 ©2009 IEEE

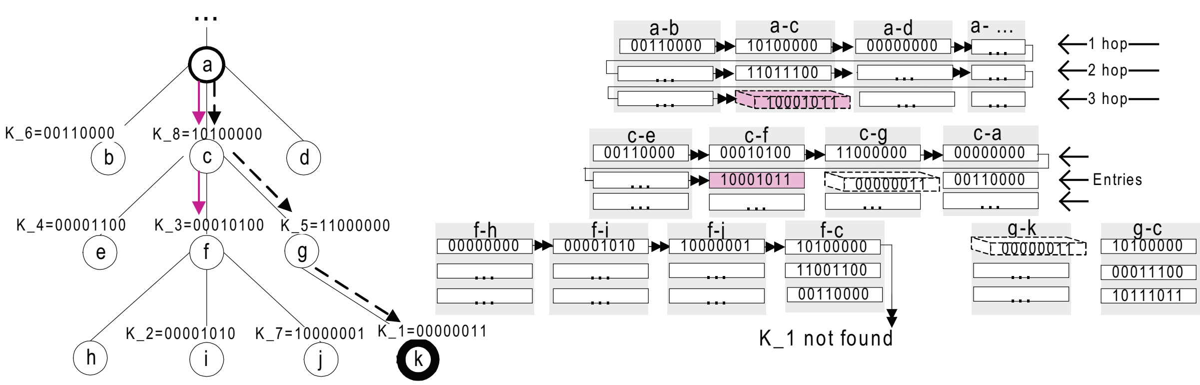

Fig. 1. Content-based routing on a regular tree topology: The 3-D dashed entries in the routing tables lead to the correct route towards locating *K_1*, while the purple ones represent the mistaken entries that were chosen by the content-based routing.

ones in the corresponding entry. When retrieving a block of stored data, routing tables are used to forward the requests hop-by-hop to a storing node. Node that receives a request checks its routing table entries row at a time starting from the one corresponding to one-hop neighbors. If the positions of the 1s in the key match the position of 1s in an entry of the routing table the key is assumed to be stored in the entry with a certain probability of false positive.

Figure 1 depicts a BF content-based routing in a regular tree network. Each node in the network stores a block of data represented with its correspondent key. Node c maintains a four-column routing table, columns *c-e*, *c-f*, *c-g* and *c-a* consisting of 3 entries each. We take *K_1* as an example of a key to search. In our illustrative example we represent with solid arrows on the tree, the route of the query sent from node a searching *K_1*. While the dashed arrows represent the correct path leading to node k storing *K_1*. We show the sequential checking of the rows in the local routing table of node a, where *K_1* is matched in the third hop of column *a-c*. As a result, the query is directed towards node c where a false positive occurs leading into forwarding the query to node f.

III. Impacts on Choosing the Number of Hash Functions

Let us denote by m the length in bits of a BF. Assume the total number of keys inserted in a BF is n, where the keys now correspond to original data blocks as well as to replicas. On the other hand, n' is the total number of distinct keys in a BF where the keys correspond only to original data blocks. This yields $n' \leq n$. Typically [1] the number of hash functions used for generating the keys is chosen to be $k = \ln 2 \frac{m}{n}$. In this section we discuss the implications of the choosing k and n on the probability of false positive.

A. Impact of replicas on k

To the best of our knowledge, none of the existing works in this area tackle the problem of choosing the optimal k in a network storing replicas. All the studies consider knowing the maximum number of elements n in a BF independently if the n elements are distinct or not. Accordingly, the values of k and m are computed by $k(n) = \ln(2)\frac{m}{n}$ such that a probability of false positive error is lower than a predefined threshold. This threshold is determined by the capability of a chosen application to support false positive errors in data localization. Our article addresses the implementations with replicas and clarify for the readers that this approximation in the choice of k is no longer the optimal for the whole network when the number of replicas increases. This is due to the fact that the number of distinct keys n' is the actual number which induces a false positive error in a filter of n elements. As long as the number of replicas is small, n' can be approximated with n and the traditional approximation is applicable. In this article we prove that this approximation is no longer valid when the number of replicas increases. This induces a considerable shifting of n' from n resulting into a poor estimation of the optimal value of k. Figure 2 illustrates an example of four data blocks to be inserted in a BF, $n = 4$. Three out of four blocks are replicated and therefore generate the same key *key-1* leading to the presence of two distinct keys in the BF, $n' = 2$. Considering a BF of 20 bits, we compute the number of hash functions that minimizes the probability of false positives. While $k(n) = \ln(2)\frac{20}{4} = 3$ for $n = 4$, we find that the number of used hash functions should in fact be $k' = \ln(2)\frac{20}{2} = 7$ for $n' = 2$.

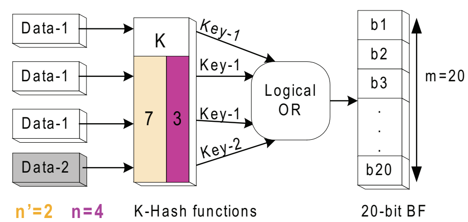

Fig. 2. Two distinct blocks of data are to be inserted in the BF. One block is replicated three times meaning that n'=2 and n=4. This leads to the use of correspondently $k = 7$ in place of 3.

B. Impact on k using the upperbound of n

In most of the applications each node receives flooded keys from a predefined neighborhood limited with a maximum

TABLE I
FALSE POSITIVE RATE UNDER VARIOUS M/N AND K COMBINATIONS.

m/n	K	$K=1$	$K=2$	$K=3$	$K=4$
2	1.39	0.393	0.400		
3	2.08	0.283	0.237	0.253	
4	2.77	0.221	0.155	0.147	0.160

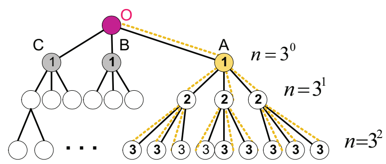

Fig. 3. Regular tree topology of degree 3 showing the nodes that are reachable from the root node O through neighbor A

number of hops, denoted by h in the following [4], [5], [6] and [7]. Upon receiving the keys, the node stores them in its routing table consisting of a matrix, with one column for each neighbor and with rows signifying the distance in number of hops to the node data is stored at. This approach leads to the highest number of n in the last rows of the matrix corresponding to the furthest hops. Therefore, the traditional adopted solution using the maximal value of n for optimizing k according to $k(n) = \ln(2)\frac{m}{n}$, minimizes the probability of error for the last hop. It is important to note that this value of k is not necessarily the best one which maximizes the probability of localizing a data in the network.

C. Impact of replicas on the BF length

Many embedded devices such as sensor nodes (i.e. TelosB [2]) have very limited size of RAM. Therefore, the major target of programmers is often to minimize the memory consumption. The impact of using n instead of n' is again introducing a major error leading to a non optimized length of BFs. Let us consider the previous example with $n = 4$ and $n' = 2$ and analyze it from the perspective of, m, the length in bits of the BF. In Table I, we show that the lowest probability of error for $\frac{m}{n} = 4$, is equal to 0.147 for $k(n) = 3$. Using $n = 4$, this probability of false positives can be achieved for $m = 16$ bits, while for $n = n' = 2$ it can be reached by using only $m = m' = 8$ bits. Taking into account the number of replicas in this small example, reduced the length of BF by 50%.

IV. PROPOSALS

This section details two proposed solutions for choosing the best value of k. The first takes into consideration the impact of the replicas and the second bypasses the problem of taking the upperbound of n. This section highlights as well a method benefiting from the presence of replicas in the network to optimize the length of the Bloom filter in use.

A. Proposal for the impact of replicas on k

Let us denote by N_{dat} the number of original data blocks. Assume q copies $[c_1, c_2, \cdots, c_q]$ of each of these blocks are

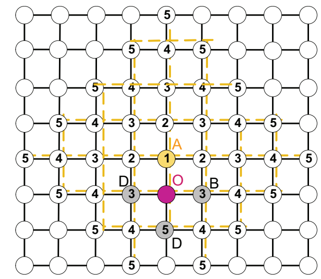

Fig. 4. Grid topology showing the nodes that are reachable from the root through neighbor A

randomly distributed among the other nodes. The total number of nodes in the network is denoted by N_{net}. Recall that n is the total number of keys in a Bloom filter and n' is the number of distinct keys in the same BF. We calculate the n' which is the real number of keys in BF that may induce a probability of error. $Ev_{d\text{-}n\text{-}r}$ is the event of having key l, $l = 1, \cdots, N_{dat}$, among the n keys in the entry. The average number of distinct keys n' in an entry of n is $\bar{n'} = N_{dat}\mathbb{P}(Ev_{d\text{-}n\text{-}r})$, where $\mathbb{P}(Ev_{d\text{-}n\text{-}r})$ is the probability of having at least one of the q copies of d in n. This yields

$$\begin{aligned}
\bar{n'} &\simeq N_{dat}\left[\frac{qn}{N_{net}} - \binom{q}{2}\frac{n(n-1)}{N_{net}(N_{net}-1)} + \right. \\
&\quad \left. \cdots + (-1)^{q+1}\binom{q}{q}\prod_{s=0}^{q-1}\left(\frac{n-s}{N_{net}-s}\right)\right] \\
&= N_{dat}\sum_{t=1}^{q}(-1)^{t+1}\binom{q}{t}\prod_{s=0}^{t-1}\left(\frac{n-s}{N_{net}-s}\right). \quad (1)
\end{aligned}$$

Having n', the value of k' that minimizes the probability of error in an entry of the routing table becomes

$$\begin{aligned}
k' &= \frac{m}{n'}\ln 2 \\
&= \frac{m\ln 2}{N_{dat}\sum_{t=1}^{q}(-1)^{t+1}\binom{q}{t}\prod_{s=0}^{t-1}\left(\frac{n-s}{N_{net}-s}\right)}, \quad (2)
\end{aligned}$$

where $\binom{q}{t}$ is the binomial coefficient.

For clarification, we consider two application examples where each node stores no more than one data block. We do not tackle the issue of border effects which is out of the scope for the paper. The storing node generates a key which identifies this block [3]. Assuming the flooded message can reach h hops, and denoting by $r = 1, \ldots, h$ the rows of the routing table, we calculate the value of n for each topology. Replacing its value in (2) determines k.

1) Regular tree application example: Let us assume a regular tree topology with degree D, the maximum number of keys in an entry of the routing table is $n = D^{r-1}$. This is equal to the number of nodes that are reachable r hops away. Figure 3 illustrates a regular tree of degree 3. The root node O has three direct neighbors and thus it maintains three routing tables. The figure shows explicitly that the number of nodes that are reachable 2 and 3 hops away from O through neighbor

A are 3^1 and 3^2. This shows that maximum 3 and 9 keys are respectively inserted in any entry of the second and the third rows of the routing table.

2) Grid topology with shortest paths: We consider a grid topology assuming the flooded messages are received via the shortest paths between the nodes. Moreover, each node is reached via a single route. The maximum number of keys inserted in an entry of the routing table is

$$n = \begin{cases} 4(r-1)-1 & \text{if } r \leq 3 \\ 4(r-1) & \text{if } 3 < r \leq h \end{cases}$$

Figure 4 shows that each node in a grid, having no border effects, has 4 direct neighbors. We illustrate a randomly chosen node O and the sets of nodes that it can reach through neighbor A. Nodes are labeled with their positions in term of number of hops relatively to node A. Considering the third hop, $r = 3$ and counting the correspondent nodes on the figure results in 7 which satisfies the first condition of our formula.

B. Proposal for the impact on k using the upperbound of n

Since we have proven that n' should be used instead of n. We further use n' in the following derivation. In order to overcome the problem of bounding n' from above two major steps are to be fulfilled. As a first step, the probability of localizing a data for any type of network needs to be formulated as a function of n. As a second step, we compute the value of n' and its corresponding optimal value of k, further in the article denoted by *k(o)*, which maximizes the probability of localizing a block of data.

As an application example, we refer to one of our previous work [8] which derives the probability of data localization in a regular tree network. We have assumed a tree network with degree N and depth D_T, where each node stores no more than one data block. Nodes on the tree were given identification numbers starting from one for the leftmost node. Moreover, each node is identified with the pair (i, j) where i, $i = 1, \cdots, D_T$, is the level of the node in the tree and j, $j = 1, \cdots, N^i$, its position in level i. We have considered N_{dat} number of original data blocks and q copies from each block $[c_1, c_2, \cdots, c_q]$. Considering $\mathbb{P}_e(n'_r) = \ln(2)\frac{m}{n'_r}$ is the probability of false positive in row r of the routing table, $r = 1, \cdots, i-1$, where n'_r is the number of distinct keys in the current entry of the routing table. With this notation we have the probability of localization

$$\begin{aligned} \mathbb{P}(Ev_{loc\text{-}d}) \approx & \sum_{i=1}^{D_T}\sum_{j=1}^{N^i}\prod_{l=0}^{q-1}\left(1 - \frac{\sum_{i'=1}^{i-1} N^{i'} + j + 1}{\sum_{i'=1}^{D_T} N^{i'} + 1 - l}\right) \\ & \times \left(\frac{q}{\sum_{i'=1}^{D_T} N^{i'} - \sum_{i'=1}^{i-1} N^{i'} - j + 1}\right) \\ & \times \prod_{i'=1}^{i-1}\left\{\left[\prod_{r=1}^{i-i'} \left(1 - \mathbb{P}_e(n'_r)\right)^N\right]\right. \\ & \left. \times \left(1 - \mathbb{P}_e(n'_{i-i'+1})\right)^{\left(\lceil \frac{j}{N^{i-i'-1}} \rceil - 1\right) \bmod N}\right\}. \end{aligned} \quad (3)$$

Adopting the same spirit of thinking, this analytical formulation can be also generalized for other network topologies. Having $\mathbb{P}(Ev_{loc\text{-}d})$ and solving the value of *k(o)* which maximizes it, would result in achieving the lowest probability of false positive in the network and the optimized probability of localizing a block of data accordingly.

C. Proposal for the impact of replicas on the BF length

Knowing the number of replicas q, the total number of nodes in the network N_{net} and n the total number of elements in the BF, we equalize equation (2) for $m = m'$ to $k = \ln(2)\frac{m}{n}$. This leads to the optimized length of BF,

$$\begin{aligned} m' &= m \times \frac{n'}{n} \\ &= m \times \frac{N_{dat} \sum_{t=1}^{q} (-1)^{t+1} \binom{t}{q} \prod_{s=0}^{t-1} \left(\frac{n-s}{N_{net}-s}\right)}{n} \end{aligned} \quad (4)$$

Using a BF length of m' bits for n' elements provides the same probability of false positive for the same k as when using a length of m bits for n elements.

V. Simulation Settings

We have verified our results on two different network topologies. A regular tree topology and a grid topology assuming shortest paths between the nodes. Complex topologies are subject of evaluation in our future work. We exclude the border effects which are not the major target of this article. We consider a grid topology of 37×37 nodes, resulting into a network of 1369 nodes. All the requests are generated from the central node. We fix the neighborhood to $h = 13$ hops, resulting into $\frac{1}{4}$ of the nodes in the network. In order to have comparable results with the previous topology, we consider the same network size for the regular tree network. Therefore, we fix the degree to 4 children and the depth to 5 hops, resulting into a network of 1365 nodes. We consider that all the requests are generated from the root node. When one considers other nodes in the network (tree or grid) as requesting nodes, the analytical as well as the simulation results are exactly the same. We fix the neighborhood to $h = 4$ hops in order to cover $\frac{1}{4}$ of the nodes in the network. In both topologies we validate the analytical results by running a set of simulations averaged over 300 independent realizations. We allow each node in the network to store at maximum one block of data. We consider Bloom filter lengths starting from 256 bits up to 896 bits. We consider 128 original blocks of data and vary the number of replicas reaching maximum the value of 10. Original blocks of data as well as their replicas are stored in randomly selected nodes.

VI. Simulation Validation

In this section we explain the simulation results that validate our two derived results for choosing the value of k which minimizes the number of false positives. We show as well the efficiency of the method benefiting from the number of replicas in minimizing the length of the Bloom filter.

TABLE II
OPTIMIZED BF LENGTH FOR 5 TIMES REPLICATED DATA IN THE GRID.

n	n'	k	m	m'	$\mathbb{P}(Ev_{loc\text{-}d})$
22	11	8	256	126	0.7254
22	11	12	384	189	0.7320
22	11	16	512	251	0.7466
22	11	20	640	314	0.7537
22	11	24	768	377	0.7605
22	11	28	896	440	0.7605

TABLE III
OPTIMIZED BF LENGTH FOR 10 TIMES REPLICATED DATA IN THE GRID.

n	n'	$k(n)$	m	m'	$\mathbb{P}(Ev_{loc\text{-}d})$
45	37	4	256	210	0.878
45	37	6	384	315	0.911
45	37	8	512	420	0.928
45	37	10	640	525	0.931
45	37	12	768	630	0.931
45	37	14	896	735	0.932

TABLE IV
OPTIMIZED BF LENGTH FOR 5 TIMES REPLICATED DATA IN THE TREE.

n	n'	$k(n)$	m	m'	$\mathbb{P}(Ev_{loc\text{-}d})$
30	27	6	256	230	0.7544
30	27	9	384	346	0.7610
30	27	12	512	460	0.7617
30	27	15	640	576	0.7618
30	27	18	768	691	0.7618
30	27	21	896	806	0.7618

TABLE V
OPTIMIZED BF LENGTH FOR 10 TIMES REPLICATED DATA IN THE TREE.

n	n'	$k(n)$	m	m'	$\mathbb{P}(Ev_{loc\text{-}d})$
60	49	3	256	208	0.8700
60	49	4	384	313	0.9272
60	49	6	512	418	0.9393
60	49	7	640	522	0.9424
60	49	9	768	627	0.9433
60	49	10	896	731	0.9435

A. Proposal validation: Impact of replicas on k

The first simulation highlights the new number of hash functions k' that shall be used instead of $k(n)$ in a network with replicas. In Tables II, III, IV and V we generate the values of n' simultaneously for the grid and the tree topologies. For the lack of space we do not compute the values of k' for the different topologies. We just note that the computation of k' is straightforward and can be generated from the four tables using $k' = \ln(2)\frac{m}{n'}$. Therefore, the smaller n' is from n, the larger is k' in comparison to $k(n)$, where $k(n) = \ln(2)\frac{m}{n}$. We take as an example n' and n from Table II. For a BF of size 384 bits, $k(n) = 12$ while the computed value of k' is 24.

B. Proposal validation: Impact on k using the upperbound of n

The aim of this simulation is to show the possibility of increasing the probability of data localization in the network by using *k(o)*, the optimal number of hash functions. We show the gain of $\mathbb{P}(Ev_{loc\text{-}d})$ when using *k(o)* in comparison with $k(n)$ that is traditionally used to optimize the latter probability for the last row of the routing table and without any consideration for the replicas. Increasing number of replicas has two opposite effects. On one hand it increases the probability of locating a data. But, on the other hand it raises the probability of false positives especially in highly connected networks. In the tree network, due to the single paths between any pair of nodes, a key stored on one of the nodes is inserted only once in the routing table of the other node. This fact helps in decreasing the number of keys in a BF and hence minimizing the probability of false positive. Conversely, the grid suffers from a multiple insertions of a key in the routing table, hence a higher probability of false positive. For this reason, we compare the percentage of enhancement between the tree and the grid. In Figures 5 and 7 each data is replicated 5 times in the network. The figures show a better enhancement of the probability of data localization for the grid more than for the tree. As an example, having a grid topology and a BF of 384 bits using a *k(o)* $= 8$ instead of $k(n) = 6$ enhances the $\mathbb{P}(Ev_{loc\text{-}d})$ of 5.9%. While using the same length of BF for the tree topology requires a *k(o)* $= 10$ instead of $k(n) = 9$ in order to reach an enhancement of 2.7%. Moreover, we increased the number of replicas to 10 in both types of networks. This leads to a 93% of storing nodes resulting into an increase in the number of keys in an entry of the routing table. Due to this fact, the probability of false positives starts to be remarkably high even with large size BFs. Therefore, we notice from Figures 6 and 8 a small room of enhancing the $\mathbb{P}(Ev_{loc\text{-}d})$ with BFs larger than 512 bits.

C. Proposal validation: Impact of replicas on the BF length

The percentage of improvement in optimizing the length of the Bloom filter highly depends on the network topology as well as on the number of replicas. A network having a high connectivity degree between the nodes provides several paths between two randomly chosen nodes. In the context of our work, this yields a key stored by one of the nodes to be inserted multiple times in the routing table the other node. The higher the connectivity degree, the more n' is shifted from n and reciprocally m' from m. Therefore, we compare the tree topology having a single path between any two nodes to the grid topology. We distribute 5 replicas for each block of data letting 46% of the nodes to store. The results of our simulation in Table II for the grid show $n' = 11$ and $n = 22$. This 50% of difference results into 50% of optimization in the length of the BF, i.e., from initially 128 bits to 62.8 bits. While these figures are relative to the grid, Table IV shows a difference of 10% between $n' = 27$ and $n = 30$ for the tree. This low difference is due to the single paths between the nodes resulting into a poor natural replication of data in the routing tables. Hence the difference between m' and m is also approximately 10% i.e., from initial 230 to 256. We highlight

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2009 proceedings.

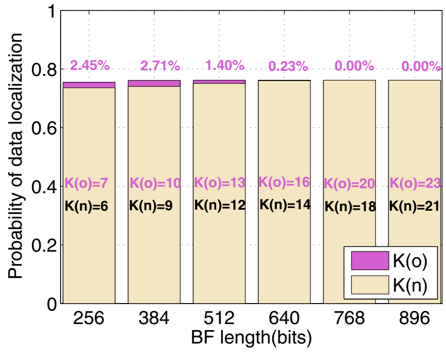

Fig. 5. The case of a tree topology having 5 replicas for each original data.

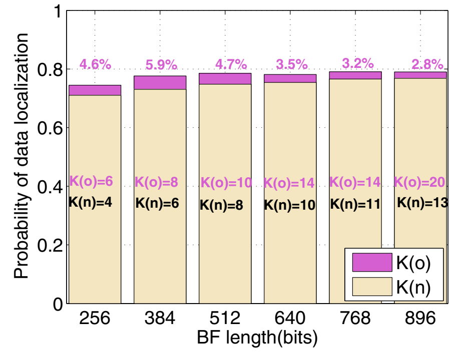

Fig. 7. The case of a grid topology having 5 replicas for each original data.

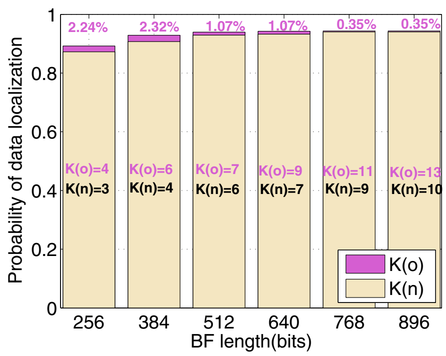

Fig. 6. The case of a tree topology having 10 replicas for each original data.

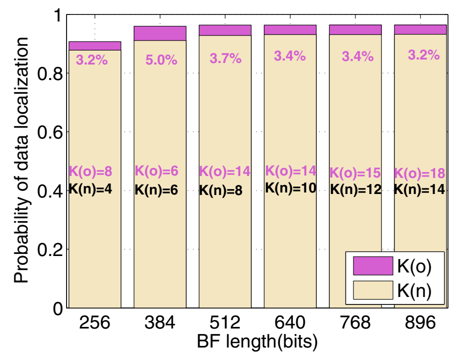

Fig. 8. The case of a grid topology having 10 replicas for each original data.

the impact of the replicas by using 10 copies for each block of data, increasing the percentage of storing nodes to 93%. For the grid topology, Table III shows $n = 45$ and $n' = 37$. The optimization of the Bloom filter length has dropped from 50% for 5 replicas to 18% for 10 replicas. This is due to the fact that the total number of keys n is an entry of the routing table, has also doubled while n' has increased 3.5 times more due to doubling the replicas as well as due to the multiple paths. In contrast the tree does not have multiple paths. Thus, we observe in Table V the values of $n = 60$ and $n' = 49$. In comparison with the results for 5 replicas we notice that n has doubled and n' has increased 1.8 times more leading to a 18% of optimization in the BF length. Finally, it is important to note from all of the four tables that the number of hash functions and the probability of data localization remain intact when using m' the optimized length of the Bloom filter.

VII. Conclusions

We have analyzed the impact of two common approaches used for choosing the number of hash functions in BFs and detailed their shortcomings if applied to networks using replicas. Moreover, we have highlighted a way of optimizing the length of the Bloom filters. We have provided two analytical expressions. The first, is the optimal number of hash functions when adding replicated elements in a BF. The second, is the optimal probability of localizing a piece of data in the network. Simulations have verified the accuracy of our analytical work. Our proposed approaches for the choices of number of hash functions optimize the probability of data localization in the network. Finally, we show an increase for the probability of data localization while reducing the length of the Bloom filter.

Acknowledgment

This work has been financially supported by the EU (Project UbiSec&Sens) and RWTH Aachen University.

References

[1] B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” in *Proc. Communications of the ACM*, vol. 13, July 1970, pp. 422–426.

[2] J. Polastre et al., “Telos: Enabling Ultra-Low Power Wireless research,” in *Proc. of IPSN’05*, CA, USA, Apr. 2005, pp. 364–369.

[3] A. Brodrei et al., “Network applications of bloom filters: A survey,” *IEEE Transactions on Communications*, vol. 30, pp. 91–99.

[4] C. Jardak et al., “Distributed Information Storage and Collection for WSNs,” in *Proc. of the fourth IEEE International Conference on Mobile Ad-hoc and Sensor Systems(MASS)*, Italy-Pisa, October 2007.

[5] M. Rupf et al., “Optimum sequence multisets for synchronous code-division multiple-access channels,” *IEEE Transactions on Information Theory*, vol. 40, pp. 1261–1266, july 1994.

[6] T. Repantis and V. Kalogeraki, “Data Dissemination in Mobile Peer-to-Peer Networks,” in *Proc. of the 6th International Conference on Mobile Data Management (ACM MDM)*, Ayia Napa, May 2005, pp. 211–219.

[7] S. C. Rhea et al., “Probabilistic location and routing,” in *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 3, Los Alamitos, June 2002, pp. 1248–1257.

[8] C. Jardak et al., “Data retrieval based on orthogonal keys in wireless sensor networks,” *Proc. of the IEEE International Conference on Communications (ICC’09)*, September 2009.