Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 7, July 2019, pp. 1947-1954
DOI: 10.23940/ijpe.19.07.p22.19471954

Efficiently Retrieving Differences Between Remote Sets using Counting Bloom Filter

Xiaomei Tian[a,b,*], Huihuang Zhao[a,b], Yaqi Sun[a,b], and Xiaoman Liang[a,b]

[a]*College of Computer Science and Technology, Hengyang Normal University, Hengyang, 421002, China*
[b]*Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, Hengyang, 421002, China*

Abstract

Retrieving differences between remote sets is widely used in set reconciliation and data deduplication. Set reconciliation and data deduplication between two nodes are widely used in various network applications. The basic idea of the difference retrieving problem is that each member of a node pair has an object set and seeks to find all differences between the two remote sets. There are many methods for retrieving difference sets, such as the standard Bloom filter (SBF), counting Bloom filter (CBF), and invertible Bloom filter (IBF). In these methods, based on the standard Bloom filter or its variants, each node represents its objects using a standard Bloom filter or other Bloom filter, which is then exchanged. A receiving node retrieves different objects between the two sets according to the received SBF, CBF, or IBF. We propose a new algorithm for retrieving differences that finds differences between remote sets using counting Bloom filters' deletion operation. The theoretical analyses and experimental results show that the differences can be retrieved efficiently. Only a very small number of differences are missing in the retrieving process, and this false negative rate can be decreased to 0% by adjusting the counting Bloom filter's parameters.

Keywords: counting Bloom filter; deletion operation; retrieving differences; data deduplication; set reconciliation

(Submitted on April 13, 2019; Revised on May 25, 2019; Accepted on June 25, 2019)

1. Introduction

In the process of set reconciliation [1-2], two remote hosts, with their respective data sets, try to get the union set of the two data sets. During the process of data deduplication [3-4], two remote hosts with each data set aim to identify the common elements of the sets and remove the repeated data or replace the repeated data with a pointer.

Both set reconciliation and data deduplication can be abstracted to efficiently find differences between two remote sets. In the reconciliation process, the difference set is used to compute the union set. In the deduplication process, the difference set is used to compute the intersection.

The most intuitive way to retrieve remote difference sets is to send all data objects in the local host directly to the remote host, and then the local host computes the difference set. This method needs to transmit all the objects in the set S_A or S_B, so the network bandwidth consumption is large and not suitable for some kinds of networks and distributed systems. A common alternative is to use update-logs with timestamps and to record the latest communication time. Here, state data such as update-logs and the latest communication time are used to compute the difference set. These methods need to maintain various state information and usually need to modify system designs before being utilized, so their applications have some limitations [5]. Additionally, there are some difference calculation methods that do not need state data [1-2, 6-13]. They can be used in various distributed network systems because they do not need to store state data or modify system designs, and they are simple and easy to use.

This paper focused on the stateless methods and aimed to find a method to retrieve all or most of the differences by

* Corresponding author.
E-mail address: tianxm@hynu.edu.cn

using the counting Bloom filter deletion operation, which is called the CBF deletion method. There are several prior works based on the standard Bloom filter or its variants, such as the **SBF method** [11], **CBF subtraction method** [12], and **IBF method** [13]. In the **SBF method**, both nodes exchange standard Bloom filters and then retrieve differences from the received Bloom filter. The IBF method utilizes invertible Bloom filters to retrieve differences between nodes. What is most similar to our method is the CBF subtraction method, which is based on the counting Bloom filter subtraction operation.

2. Algorithm of Retrieving Differences based on Counting Bloom Filters Deletion Operation

A Bloom filter [14] is an m-bit vector and a set of k hash functions $h_1, h_2, \cdots, h_k$. The m bits of the vector are initialized to 0s. Each element in the set S is hashed into k corresponding bits in the bit vector with k hash functions, and these bits are thereby set to 1. In order to test whether x is in S or not, x's hash addresses $h_1(x), h_2(x), \cdots, h_k(x)$ are computed at first, and then the bit vector is checked whether all k bits $(h_1(x), h_2(x), \cdots, h_k(x))$ are 1s. If they are all 1s, then x is determined to belong to S. Otherwise, as long as any of the k bits is 0, x must not belong to S. It is possible to misjudge an element y that is not in S as belonging to S. The element y is called a false positive.

The probability of a false positive in a Bloom filter for a set S is $(1-e^{-k|S|/m})^k$. Counting the Bloom filter [15] expands every bit of the standard Bloom filter to a counter of r bits. Moreover, the counting Bloom filter can support deletion operations, while the standard Bloom filter cannot.

For the convenience of description, the remote difference retrieving process based on the counting Bloom filter deletion operation is abbreviated as **CBF deletion method** (shown in Figure 1). The specific procedures are as follows.

Step 1 Host B sends $CBF(S_B)$ to host A.

Step 2 Host A queries $CBF(S_B)$ whether the element x in S_A belongs to S_B according to the query rule of counting Bloom filters.

Step 3 If the retrieval result finds that x belongs to S_B, x is deleted from $CBF(S_B)$, and it is judged as not belonging to the difference set S_A - S_B.

Step 4 If the retrieval result finds that x is not a member of S_B, it is determined that x belongs to the difference set S_A - S_B.

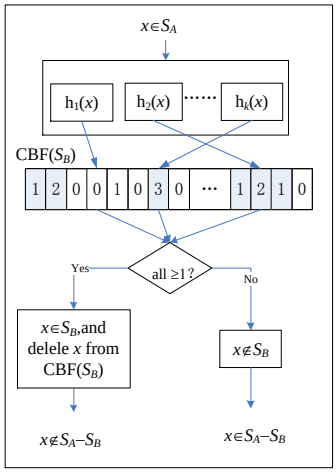


Figure 1. Algorithm of retrieving differences based on counting Bloom filters deletion operation

3. Performance Analyses

For data sets S_A and S_B, it is assumed that $|S_A| = n_1$, $|S_B| = n_2$, and $|S_A \cap S_B| = n_0$. $CBF(S_A)$ and $CBF(S_B)$ are counting Bloom filters for S_A and S_B respectively. They are vectors composed of m counters that use r binary bits, and they utilize the same k hash functions.

For ease of understanding and presentation, the notations used for subsequent discussions are given, as shown in Table 1.

Table 1. Notations and their meanings

Notations	Meanings	Notations	Meanings
n_1	the size of set S_A	k	the number of hash functions
n_2	the size of set S_B	$BF(S_B)$	standard Bloom filter for S_B
n_0	the size of set $S_A \cap S_B$	$CBF(S_B)$	counting Bloom filter for S_B
d_1	the size of set S_A - S_B	$IBF(S_B)$	invertible Bloom filter for S_B
d_2	the size of set S_B - S_A	$CBF(S_B - S_A)^{del}$	the vector after deleting S_A from $CBF(S_B)$
d	$d_1 + d_2$	$CBF(S_B - x)^{del}$	the vector after deleting x from $CBF(S_B)$
m	the length of Bloom filer vector	$CBF(S_B - X)^{del}$	the vector after deleting X from $CBF(S_B)$

3.1. False Negatives

For the CBF deletion method, the false negative ratio is defined as follows:

$$FN_{del} = \frac{\left|\left\{x \middle| x \in S_{\Delta}\right\}\right|}{\left|S_A - S_B\right|} \tag{1}$$

Where S_Δ is a subset of S_A - S_B and the objects in which are deleted by the CBF deletion method.

Since the deletion operation is used while retrieving differences, the number of non-zero counters in the counting Bloom filter $CBF(S_B - X)^{del}$ decreases as the deleted intersection elements increase, which reduces false positives of querying S_A elements by $CBF(S_B - X)^{del}$. Thus, there will be fewer S_A - S_B elements misjudged in S_B. For difference retrieval, fewer S_A - S_B elements are misjudged to be not in S_A - S_B, that is, the false negative probability becomes smaller during the process of deletion operation. Therefore, the false negative rate of the CBF deletion method is related to the deletion sequence of intersection elements.

Theorem 1 There are false negatives in the remote difference retrieval algorithm based on the counting Bloom filter deletion operation: the false negative probability under the best condition is $(1-e^{-k(n_2-n_0)/m})^k$, the false negative probability under the worst condition is $(1-e^{-kn_2/m})^k$, and the false negative probability is

$$\frac{1}{n_2 - n_0} \sum_{i=1}^{n_2-n_0} \left(1 - e^{-k\left(n_2 - \frac{in_0}{n_2-n_0}\right)/m}\right)^k$$

when the intersection elements are randomly and evenly distributed in S_B.

Proof Due to the inherent misjudgment of counting Bloom filters, the following phenomenon occurs when $CBF(S_B)$ is used to query S_A elements: elements that are not in S_B are misjudged to be elements in S_B, so there will be a small number of objects in S_A - S_B judged as elements in S_B. This small number of S_A - S_B elements are determined to be not in S_A - S_B, that is, they are false negatives. In other words, the difference retrieving algorithm based on the counting Bloom filters deletion operation will have a small number of false negatives during the retrieval operation.

In the best case, all objects in $S_A \cap S_B$ are in the front of S_A. When retrieving these intersection elements and then removing them from $CBF(S_B)$, there will be no false negatives. After deleting all the intersection elements, we use $CBF(S_B - \{S_A \bigcap S_B\})^{del}$ to retrieve differences, and the false negative probability is $(1-e^{-k(n_2-n_0)/m})^k$.

In the worst case, all elements in $S_A \cap S_B$ are in the end of S_A. False negatives appear when S_A - S_B elements are retrieved from $CBF(S_B)$, and the false negative probability is $(1-e^{-kn_2/m})^k$. No false negatives will arise when deleting the subsequent intersection elements from $CBF(S_B)$. Therefore, the total false negative probability of the algorithm in the worst case is $(1-e^{-kn_2/m})^k$.

When $S_A \cap S_B$ elements are randomly and evenly distributed in S_A, the false negative probability when retrieving the first element of S_A - S_B in S_A is

$$\left(1-e^{-k\left(n_2-\frac{n_0}{n_2-n_0}\right)/m}\right)^k$$

the false negative probability when retrieving the second element of S_A - S_B in S_A is

$$\left(1-e^{-k\left(n_2-\frac{2n_0}{n_2-n_0}\right)/m}\right)^k$$

the false negative probability when retrieving the i^{th} element of S_A - S_B in S_A is

$$\left(1-e^{-k\left(n_2-\frac{in_0}{n_2-n_0}\right)/m}\right)^k$$

and the false negative probability when retrieving the last element of S_A - S_B in S_A is $\left(1-e^{-k(n_2-n_0)/m}\right)^k$. Therefore, we can get the false negative probability average of the algorithm:

$$\frac{1}{n_2-n_0}\sum_{i=1}^{n_2-n_0}\left(1-e^{-k\left(n_2-\frac{in_0}{n_2-n_0}\right)/m}\right)^k$$

According to Theorem 1, in the worst case, the false negative probability of the remote difference retrieval algorithm based on the counting Bloom filter deletion operation is related to n_2, and it is the same as the false positive probability of retrieving the elements in S_B from $CBF(S_B)$.

3.2. False Positives

False positives of the remote difference retrieval algorithm based on the counting Bloom filter deletion operation refer to the phenomenon that elements in S_A but not in S_A - S_B (i.e., $S_A \cap S_B$ elements) are determined to be in S_A - S_B according to the counting Bloom filter $CBF(S_B)$. These false positives are different from the false positives of the Bloom filter [14-15]: false positives of the Bloom filter are elements not in S that are judged to be elements of S based on the Bloom filter $BF(S)$.

The false positive calculation formula of the remote difference retrieval algorithm based on the counting Bloom filter deletion operation is as follows:

$$FP_{del} = \frac{\left|\left\{x \middle| x \in S_{\cap}\right\}\right|}{\left|S_A \bigcap S_B\right|} \tag{2}$$

Where $S_{\cap}$ is a subset of $S_A \cap S_B$ and the objects in which cannot be deleted from $CBF(S_B - X)^{del}$.

Theorem 2 False positives may exist in the retrieval process of the remote difference retrieval algorithm based on the counting Bloom filters deletion operation: in the best case, there are no false positives; the false positive probability under the worst case is

$$\frac{(n_2-n_0)\times\left(1-e^{-kn_2/m}\right)^k \times\left(\sum_{i=1}^{k} i\times\binom{k}{i}\left(e^{-k(n_2-1)/m}\right)^i\left(1-e^{-k(n_2-1)/m}\right)^{k-i}\right)}{n_2}$$

and the false positive probability is

$$\frac{1}{n_0}\times\sum_{j=1}^{n_0}\left(\frac{n_2-n_0}{n_0}\times\frac{n_2-j}{n_2}\times\left(1-e^{-k(n_2-j)/m}\right)^k\times\left(\sum_{i=1}^{k}i\times\binom{k}{i}\left(e^{-k(n_2-1)/m}\right)^i\left(1-e^{-k(n_2-1)/m}\right)^{k-i}\right)\right)$$

when the intersection elements are randomly and evenly distributed in S_B.

Proof According to the description of the CBF deletion method, for an element x in S_A, if $x\in S_A\cap S_B$, according to the decision rule of the algorithm, there may be an element $y\in S_A$. Its hash address $h_1(y)$, $h_2(y)$, ⋯, $h_k(y)$ partially coincides with the hash address of x $h_1(x)$, $h_2(x)$, ⋯, $h_k(x)$, that is, there are $1<=i, j<=k$, which makes $h_i(y) = h_j(x)$. If querying y occurs before querying x, according to $CBF(S_B-X)^{del}$, the following result may be obtained: $y\in S_B$, and then y is deleted from $CBF(S_B-X)^{del}$. This deletion operation may clear the $h_j(x)^{\text{th}}$ counter, thereby affecting the correctness of querying x. In other words, in the difference retrieval process based on the counting Bloom filter deletion operation and for elements in S_A that do not belong to S_A - S_B, if they are queried with $CBF(S_B-X)^{del}$ after continuous deletion, they may be judged to belong to the difference set S_A - S_B (that is, false positives exist).

In the best case, the elements in $S_A\cap S_B$ are located at the front of S_A. According to the CBF deletion method, if $x\in S_A\cap S_B$, it is inevitable to delete x from $CBF(S_B)$ successfully and conclude that $x\notin S_A$ - S_B. That is to say, if $x\in S_A\cap S_B$, the difference retrieval algorithm proposed in this paper is bound to draw the conclusion that $x\notin S_A$ - S_B, and there are no false positives.

In the worst case, the elements in $S_A\cap S_B$ are located at the rear of S_A. According to Theorem 1, there will be a certain probability of misjudging the elements in S_A - S_B as belonging to S_B and deleting them from $CBF(S_B)$. This probability p_{mis} is $(1-e^{-kn_2/m})^k$. In [16], Guo et al. pointed out that this error deletion would cause subsequent misjudgment that S_B elements (i.e., elements in $S_A\cap S_B$) are judged to not be in S_B. Each element in S_A - S_B that is misjudged and deleted will lead to the misjudgment of S_B elements. The expected value of misjudgment probability is:

$$E_0=\sum_{i=1}^{k}i\times\binom{k}{i}\left(e^{-k(n_2-1)/m}\right)^i\left(1-e^{-k(n_2-1)/m}\right)^{k-i} \tag{3}$$

Thus, false positives of the difference retrieval caused by the misjudged and deleted S_A - S_B elements can be obtained as follows:

$$E_{fp}=(n_2-n_0)\times p_{mis}\times E_0\times\frac{n_0}{n_2}$$

Finally, in the worst case, the false positive probability of the difference retrieval is expected to be as follows:

$$FP_{del}=\frac{E_{fp}}{n_0}=\frac{(n_2-n_0)\times p_{mis}\times E_0}{n_2}=\frac{(n_2-n_0)\times\left(1-e^{-kn_2/m}\right)^k\times\left(\sum_{i=1}^{k}i\times\binom{k}{i}\left(e^{-k(n_2-1)/m}\right)^i\left(1-e^{-k(n_2-1)/m}\right)^{k-i}\right)}{n_2} \tag{4}$$

In the random case, the elements in $S_A\cap S_B$ are scattered in S_A uniformly and randomly, and when the j^{th} element in $S_A\cap S_B$ is queried, a certain amount of the elements in S_A - S_B are deleted from $CBF(S_B)$ by mistake. This amount is $D_{mis}^j=\frac{n_2-n_0}{n_0}\times\left(1-e^{-k(n_2-j)/m}\right)^k$. The count of false positives caused by these S_A - S_B elements, which are deleted by mistake, is expected to be:

$$E_{fp}^j=D_{mis}^j\times E_0\times\frac{n_0-j}{n_2}=\frac{n_2-n_0}{n_0}\times\frac{n_0-j}{n_2}\times\left(1-e^{-k(n_2-j)/m}\right)^k\times\left(\sum_{i=1}^{k}i\times\binom{k}{i}\left(e^{-k(n_2-1)/m}\right)^i\left(1-e^{-k(n_2-1)/m}\right)^{k-i}\right) \tag{5}$$

Thus, under random conditions, the false positive probability of the difference retrieval is:

$$FP_{del} = \frac{1}{n_0} \times \sum_{j=1}^{n_0} E_{fp}^{j} = \frac{1}{n_0} \times \sum_{j=1}^{n_0} \left(\frac{n_2 - n_0}{n_0} \times \frac{n_0 - j}{n_2} \times \left(1 - e^{-k(n_2 - j)/m}\right)^k \times \left(\sum_{i=1}^{k} i \times \binom{k}{i} \left(e^{-k(n_2-1)/m}\right)^i \left(1 - e^{-k(n_2-1)/m}\right)^{k-i} \right) \right) \quad (6)$$

4. Simulation Experiments

Simulation experiments are carried out in this section to test the actual effects of the algorithm. Our method is compared with the existing SBF method [11], CBF subtraction method [12], and IBF method [13], which are similar to our method to a certain extent.

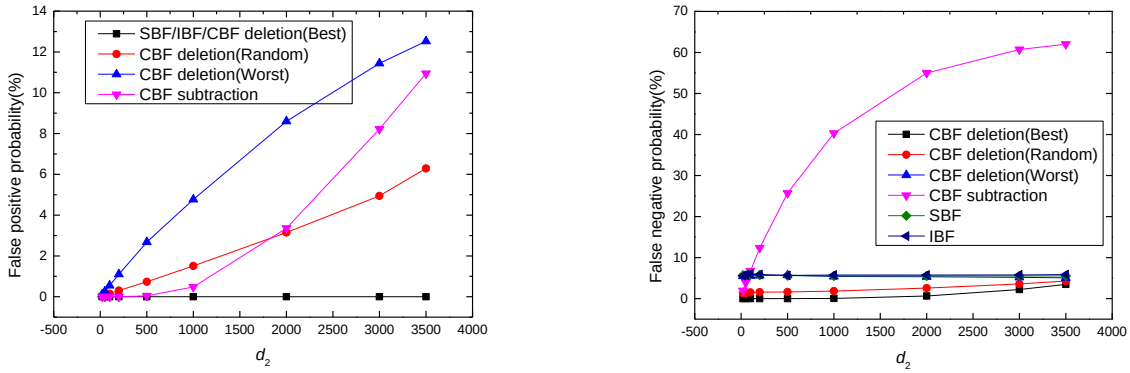
4.1. Experimental Setting

For the CBF subtraction method and CBF deletion method, we use the CBF implementations in [12] and extend them to support difference retrieval. For the SBF method, we implement the Bloom filter structure [14]. For the IBF method, the invertible Bloom lookup table structure is implemented [13].

In our experiments, we use BKDRHash, APHash, DJBHash, JSHash, RSHash, and other functions as hash functions. Each CBF, SBF, and IBF utilizes four hash functions since such a setting is sufficient to make Bloom filters work well. The set sizes of S_A and S_B are fixed to 4000, and the difference set sizes d_2 are 20, 50, 100, 200, 500, 1000, 2000, and 3000. Under these circumstances, d_1 is equal to d_2. The common elements of S_A and S_B are placed in three different positions: at the front of S_A or S_B (the best case), at the end of S_A or S_B (the worst case), or randomly and evenly distributed in S_A or S_B (the random case). For all comparative algorithms, the lengths of Bloom filter structures are 24000. For each size of the difference set, we report the average results among 100 rounds of experiments.

4.2. Results and Discussion

In all cases, the false positive probabilities of the IBF method and SBF method are always 0, the false negative probabilities of both methods are almost the same, and the corresponding curves of the IBF method and SBF method coincide overall in Figure 2.



(a) False positive probability (b) False negative probability

Figure 2. False positive probability and false negative probability in best, random, and worst cases

As can be seen from Figure 2, the false positive probabilities and the false negative probabilities of the IBF method, SBF method, and CBF subtraction method are independent of the location of intersecting elements. The CBF deletion method shows significantly different performances in three different situations. In the best case, the CBF deletion method has no false positives, and its false negative probability is superior to the other three algorithms. In the random case, the CBF deletion method has the least false negatives of all comparative algorithms, but there are certain false positives. Compared with the CBF subtraction method, the CBF deletion method has a significant advantage in false negatives when the difference set size is large. In the worst case, the CBF deletion method has the largest false positive rate, and its false negative rate degenerates to the same level as those of the IBF method and SBF method.

Moreover, false positive probabilities and false negative probabilities of the IBF method and SBF method are independent of the size of the difference set d_2. The CBF subtraction method shows less false negatives than the IBF method and SBF method when d_2 is very small (such as less than 100). The false negative probability of the CBF subtraction method increases extremely fast with the growth of d_2 when d_2 is greater than 100. When d_2 is 3500, the false negative probability of the CBF subtraction method reaches 62%, which is completely incompetent to solve the difference retrieval problem. The false positive probability of the CBF subtraction method increases slowly with an increase in d_2. The false positive probability of the CBF deletion method is 0 in the best case, increases more rapidly with the growth of d_2 in the random case and worst case, and the false negative probability of the CBF deletion method does not increase (in the worst case) or increases slightly with the growth of d_2 (in random case and best case).

5. Conclusions

Among the above typical difference retrieval algorithms, our method deletes intersection elements constantly during the retrieval process, resulting in more zero counters and less non-zero counters in counting Bloom filters. Therefore, compared with our method, the false negative probabilities of the difference retrieval algorithm based on the standard Bloom filter and the difference retrieval algorithm based on the invertible Bloom filter are higher, and the false negative probability of the difference retrieval algorithm based on the counting Bloom filter subtraction operation is too sensitive to d_2 and only applicable to small d_2.

In the best case, the difference retrieval algorithm based on the counting Bloom filter deletion operation does not have false positives and only very few false negatives. When the difference set size is small, the false negative rate can be neglected by using appropriate parameter settings, and then it can be directly used to solve the remote difference retrieval problem. Furthermore, it can be used for set reconciliation and data deduplication.

In this paper, the deletion operation of the counting Bloom filter is used to complete the retrieval of sets' differences. It has the advantages of simple implementation, high retrieval efficiency, and accurate results, and it can be widely used in data reconciliation areas and data deduplication areas such as mobile data synchronization systems and storage systems for big data.

Acknowledgements

This work is supported by the National Science Foundation of China (No. 61503128, 61772182, 61772178), Science and Technology Plan Project of Hunan Province (No. 2016TP1020), Open Fund Project of Hunan Provincial Key Laboratory of Intelligent Information Processing and Application for Hengyang Normal University, Scientific Research Fund of Hunan Provincial Education Department (No. 16C0226), and Hunan Natural Science Foundation (No. 2017JJ4012).

References

1. J. Byers, J. Considine, and M. Mitzenmacher, "Fast Approximate Reconciliation of Set Differences," BU Computer Science TR, Boston University, 2002
2. D. Starobinski, A. Trachtenberg, and S. Agarwal, "Efficient PDA Synchronization," *IEEE Transactions on Mobile Computing*, Vol. 2, No.1, pp. 40-51, 2003
3. B. Zhu, K. Li, and H. Patterson, "Avoiding the Disk Bottleneck in the Data Domain Deduplication File System," in *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, pp. 1-14, San Jose, California, USA, February 2008
4. P. Kulkarni, F. Douglis, J. Lavoie, and J. M. Tracey, "Redundancy Elimination Within Large Collections of Files," in *Proceedings of the 2004 USENIX Annual Technical Conference*, pp. 1-14, Boston, MA, USA, June 2004
5. D. Eppstein, M. T. Goodrich, F. Uyeda, and G. Varghese, "What's the Difference? Efficient Set Reconciliation Without Prior Context," in *Proceedings of the ACM SIGCOMM 2011*, pp. 218-229, Toronto, Ontario, Canada, August 2011
6. M. Skjegstad and T. Maseng, "Low Complexity Set Reconciliation using Bloom Filters," in *Proceedings of the 7th ACM SIGACT/ SIGMOBILE International Workshop on Foundations of Mobile Computing*, pp. 33-41, San Jose, California, USA, June 2011
7. Y. Minsky, A. Trachtenberg, and R. Zippel, "Set Reconciliation with Nearly Optimal Communication Complexity," *IEEE Transactions on Information Theory*, Vol. 49, No. 9, pp. 2213-2218, 2003
8. A. Trachtenberg, D. Starobinski, and S. Agarwal, "Fast PDA Synchronization using Characteristic Polynomial Interpolation," in *Proceedings of IEEE INFOCOM 2002*, pp.1510-1519, New York, NY, USA, 2002
9. Y. Minsky and A. Trachtenberg, "Practical Set Reconciliation," BU Computer Science TR, Boston University, 2002
10. J. W. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed Content Delivery across Adaptive Overlay Networks," *ACM SIGCOMM Computer Communication Review*, Vol. 32, No. 4, pp. 47-60, 2002
11. Z. Y. Hu, X. Q. Teng, D. Guo, B. B. Ren, P. Lv, and Z. Liu, "Comparing Set Reconciliation Methods based on Bloom Filters and Their Variants," *Tsinghua Science and Technology*, Vol. 21, No. 2, pp. 157-167, 2016

12. D. Guo and M. Li, "Set Reconciliation via Counting Bloom Filters," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 10, pp. 2367-2380, 2013
13. R. Gabrys and A. Coker, "Set Reconciliation in Two Rounds of Communication," in *Proceedings of International Command and Control Research and Technology Symposium* (*ICCRTS*), pp. 1-7, Annapolis, Maryland, USA, June 2015
14. B. H. Bloom, "Space / Time Trade-offs in Hash Coding with Allowable Errors," *Communications of the ACM*, Vol. 13, No. 7, pp. 422-426, 1970
15. L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary Cache: a Scalable Wide-area Web Cache Sharing Protocol," *IEEE/ACM Transactions on Networking* (*TON*), Vol. 8, No. 3, pp. 281-293, 2000
16. D. Guo, Y. H. Liu, X. Y. Li, and P. L. Yang, "False Negative Problem of Counting Bloom Filter," *IEEE Transactions on Knowledge & Data Engineering*, Vol. 22, No. 5, pp. 651-664, 2010

Xiaomei Tian received her Ph.D. in 2013 from Hunan University. She is currently a professor at Hengyang Normal University. Her main research interests include trusted systems and networks.

Huihuang Zhao received his Ph.D. in 2010 from Xidian University. He is currently an associate professor at Hengyang Normal University. His main research interests include compressive sensing, machine learning, and image processing.

Yaqi Sun received her M.Sc. degree in 2013 from Guilin University of Technology. Her main research interests include character recognition and image processing.

Xiaoman Liang is a professor at Hengyang Normal University. His main research interests include wireless networks and machine learning.