7659, England, UK
Journal of Information and Computing Science
Vol. 2, No. 1, 2007, *pp*. 55-60

A Routing Mechanism by Distance-weighted Bloom Filter [*]

Xun Duan [+] and Jian-shi Li

School of Computer Science & Engineering, Guizhou University, Guiyang, P. R. China.

(*Received August 31, 2006, Accepted November 2, 2006*)

Abstract: Bloom filters are used for resource routing and distance-weighted Bloom filters (dwBFs) are used as a concise representation of routing information for scattered resources in overlay networks. Today's DHT-based P2P lookups are always influenced by unnecessarily long routers in the underlay network. The reason is that the false positives appearing in DHT lookup can lead to incorrect direction. To apply dwBFs to scattered resources in overlay network can form a routing correcting mechanism. It can reduce the possibilities of incorrect direction under cases of false positives. To use dwBFs also can resolve another problem with DHT, its ineffectiveness in replica-handle. This paper describes a routing algorithm based on dwBFs and shows the results of simulation experiment.

Keywords: Bloom filters, Distributed Hash Table (DHT), overlay network, distance-weighted Bloom filter(dwBF)

1. Introduction

Today's exponential growth in network bandwidth and storage capacity has inspired a whole new class of distributed, peer-to-peer storage infrastructures. Systems such as Farsite, Freenet , OceanStore, CFS, and PAST seek to capitalize on the rapid growth of resources to provide inexpensive, highly-available storage without centralized servers. The designers of these systems propose to achieve high availability and long-term durability in the face of individual component failures through replication and coding techniques.

Although wide-scale replication has the potential to increase availability and durability, it introduces two important challenges to system architects. First, if replicas may be placed anywhere in the system, how should we locate them? Second, once we have located one or more replicas, how should we route queries to them? We can formulate the combination of these two operations as a single location and routing problem that efficiently routes queries from a client to the closest replica adhering to certain properties, such as the replica with the shortest network path to the client or the replica residing on the least loaded server. In many cases, combining location and routing into a single, compound operation yields the greatest flexibility to route queries quickly with minimal network overhead. The importance of such location-independent routing techniques is well recognized in the community, and several proposals such as CAN , Chord, Pastry , and Tapestry are currently under study.

A Bloom filter[1] is a simple space-efficient randomized data structure for representing a set in order to support membership queries. The space efficiency is achieved at the cost of a small probability of false positives, but often this is a convenient trade-off. Although Bloom filters were invented in the 1970's and have been heavily used in database applications, they have only recently received widespread attention in the networking literature. Peer-to-peer applications are a natural place to use Bloom filters, as collaborating peers may need to send each other lists of URLs, packets, or object identifiers.

A distance-weighted Bloom filter(dwBF) is a cluster of integers instead of a cluster of bits in standardized Bloom filter. We introduce a routing mechanism based on dwBF.

Many P2P search mechanism provide the basic rooter structures unrelated to local storage. Some of them are DHT-based[2]. Some researchers begin to be concerned about the problem of overlay dilation. Among them, Chord[2] uses an extra finger set connected to nodes in close proximity. Either Pastry[3] or Tapestry[4] uses geometrical distance as the basis to decide the rooter.

[*] Project supported by the National Science Foundation of China under Grant No.60463001

[+] Xun Duan. *Tel.*: +86-139 8504 6881; fax: +86-851-362 7649
E-mail address: Cse.xduan@gzu.edu.cn and danspray@126.com

Now, there are a lot of proposals about the rooter structures based on content or name. Here are some brief introductions of them. For example, W. Adjie-Winoto[5] proposes International Naming System(INS). INS allows the search to be very flexible, yet it lacks the capacity of being extended. Gritter[6] designs in Internet the frame for content rooter in which rooter proceeds in a way similar to BGP. This puts naming system and rooter into close connection, supporting replication naturally. As for extension, based on naming set, it takes a structuralized naming form.

Bloom filter was applied in data base at earlier time, but more in Internet now. In the aspect of rooter, Czerwinski[7] provides a general framework as part of resource search service. Hsiao[8] also uses Bloom filter in other geographical rooter systems to serve mobile computers. In this work, space is subdivided into smaller spaces of tree structures. Information packages are transmitted along the trees. Rhea and ubiatowicz[9] design a peer-to-peer way to locate the files. In overlay network, each peer keeps an attenuated Bloom filter. An attenuated Bloom filter is an array of Bloom filters for each distance d. Here, d can reaches some maximum value d_{max} which is relatively small, and the d-th Bloom filter keeps track of files reachable through the edge by exact d hops on the overlay network. This way can reduce greatly the overlay delay needed in searching for files nearby, but it has less effect on files at a distance.

The remainder of this paper is organized as follows: Section2 describes Bloom filter, distance-weighted Bloom filter and its routing algorithm. Section 3 shows the conclusion and discussion.

2. Bloom Filter for Routing

2.1. Bloom Filters

Bloom filters[10] are an efficient, lossy way of describing sets. A Bloom filter is a bit-vector of length n with a family of independent hash functions, each of which maps from elements of the represented set to an integer in [0;n). To form a representation of a set, each set element is hashed, and the bits in the vector associated with the hash functions' results are set. A Bloom filter for representing a set $S=\{x_1, x_2, \Lambda\ , x_n\}$ of n elements is described by an array of m bits, initially all set to 0.

A Bloom filter uses k independent hash functions with range [1..m]. We make the natural assumption that these hash functions map each item in the universe to a random number uniform over the range [1..m] for mathematical convenience. (In practice, reasonable hash functions appear to behave adequately.) For each element , the bits are set to 1. A location can be set to 1 multiple times, but only the first change has an effect. To check if an item y is in S, we check whether all are set to 1. If not, then clearly y is not a member of S. If all bits are set to 1, we assume that y is in S, although we are wrong with a non-zero probability. Hence a Bloom filter may yield a false positive, where it suggests that an element y is in S even though it is not. For example, y does not exist in S but some elements $w, \cdots, z$ in S covers all.

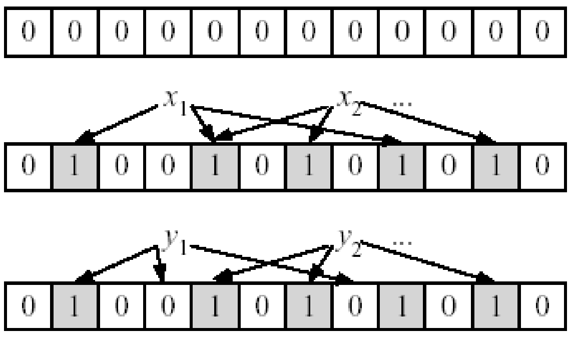

Figure 1: Bloom filter

Figure 1 is an example of a Bloom filter. The filter begins as an array of all 0's. Each item x_i in these S is hashed k times, with each hash yielding a bit location; these bits are set to 1. To check if an element y is in the set, hash it k times and check the corresponding bits. The element y_1 cannot be in the set, since one of the bits is a 0. The element y2 is either in the set or it is a false positive.

The false positive rate of a Bloom filter is a well-defined, linear function of its width m, the number of hash functions k and the cardinality of the represented set n, namely $m = \alpha n$. Here, α is called the load factor. For many applications, false positives may be acceptable as long as their probability is sufficiently small. For example, when $\alpha = 8$, the false positive rate is about 2%.

2.2. Routing by Distance-weighted Bloom Filter

Firstly, we discuss how to route in an ordinary network. Ordinary here means that there is no limitation to any topology. It can be a tree or anything else and it can be logical or physical. Bloom filter cannot be used in routers based on address such as IP, which can be more suited in other routers such as RIP, OSPF and BGP. Bloom filter takes advantage under the condition that no aggregation is possible.

In order to realize better how universal router is, we put the notion of distance into Bloom filter. A distance-weighted Bloom filter is a cluster of integers instead of a cluster of bits in standardized Bloom filter. A unit is called a bit in standardized Bloom filter but a slot in dwBF. So hash function reflects the unit set into each slot of the cluster. The distance here, similar to IP, is counted by hops or line delays. Suppose a function allows a node to decide the distance to the peer. The size of each slot is big enough to store the longest path in the Internet, namely the Internet diameter. The nodes in the Internet exchange dwBF in a way similar to RIP. Unlike the traditional way of calculating, the distance is used in slots, not in nodes, Internet or files. Each node keeps a dwBF for a peer. The dwBF kept describes the files set got through the peer and the estimated distance to the address of the files. When the dwBF changed, the node informs its peer to adjust the corresponding dwBF. Suppose the node P serves S_p, a file set, and has a peer set, P_p. V_{pq} describes the dwBF from p to q. $v_{pq}(i)$ is the i slot. When i=1,….,m, and $\delta_{p,q}$ is the distance from p to q. h_j, m and k is the same as in standardized Bloom filter. The procedure of getting V_{pq} is as follows:

Procedure 1 describes the construction of dwBF. Procedure 2 describes the routing by dwBF.

Procedure 1:

```
1．for each $x$ belongs to $S_p$， repeat 2、3
2．  for j = 1 to k
3．    if $h_j(x) = 1$， then
          $v_{p,q}(i) = 1$
          goto 12
4．for each $r$ belongs to $P_p$， repeat 5
5．  if $v_{r,p}(i) \neq 0$ then
        goto 8
6．$v_{p,q}(i) = 0$
7．repeat 12
8．$m = -\infty$
9．for each $r$ belongs to $P_p$ and $r \neq q$， repeat 10
10．  $m = \min(m, v_{r,p}(i))$
11．  $v_{p,q}(i) = m + \delta_{p,q}$
12．end
```

When peers search file x on node p. Node p firstly checks its local resource. If the file is not available locally, node p checks the dwBFs sent by its peers to choose the next hop by the following procedure 2.

Procedure 2：

```
1．for j = 1 to k
2．  if $v_{r,p}(h_j(x)) = 0$
       ignore peer $q$
       goto 17
3．for each $q$ belongs to $P_p$， repeat 4－8
```

```
4.  $e_q = -\infty$
5.  for j = 1 to k
6.    if $v_{q,p}(h_j(x)) > e_q$, then
      $e_q = v_{q,p}(h_j(x))$
7.  $c_q = 0$
8.  for j = 1 to k
          if $v_{q,p}(h_j(x)) = e_q$, then
              $c_q = c_q + 1$
9.  $m = +\infty$
10. for each $q$ belongs to $P_p$, repeat 11
11.   $m = \min(m, e_q)$
12. $n = -\infty$,
13. for each $q$ belongs to $P_p$, repeat 14
14.   if $e_q = m$, then
      $n = \max(n, c_q)$
15. for each $q$ belongs to $P_p$, repeat 16
16.   if $e_q = m$ and $c_q = n$, then
      choose peer $q$
      goto 16
17. End
```

e_q is an estimate of the distance to x via q, and c_q is the number of slots $v_{q;p}$ that give e_q

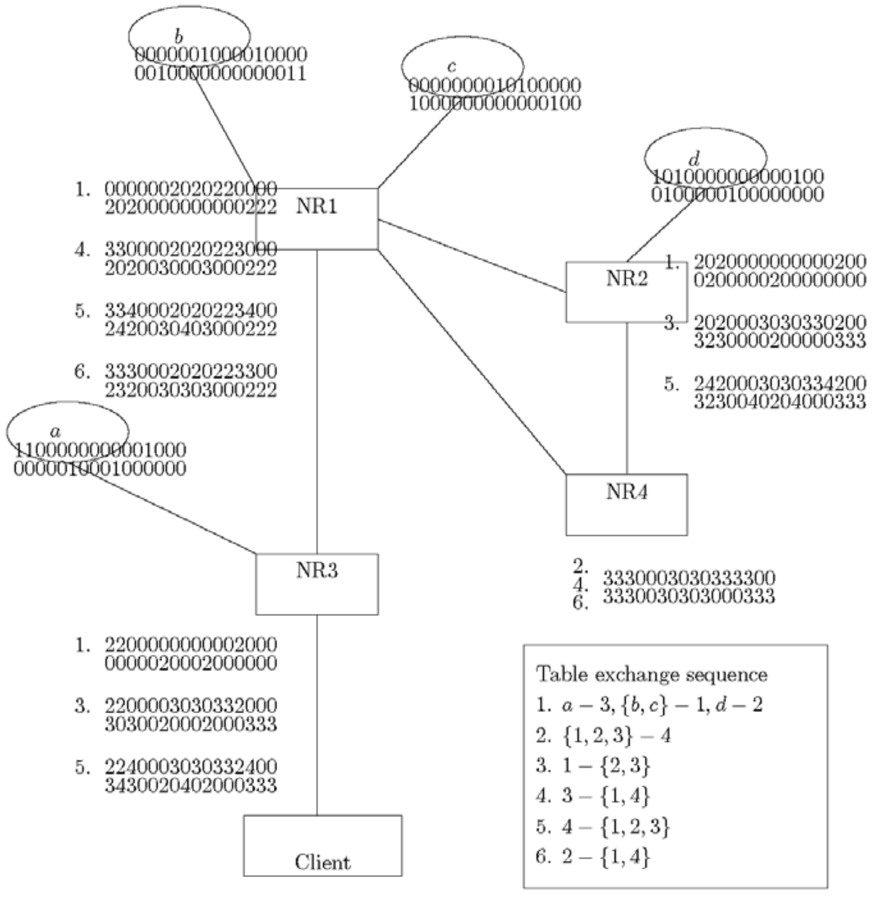

Figure 2: Routing process

Figure 2 shows a routing process by dwBF.

The second step in procedure 2 is related to the false location. First, suppose there is only a file x, and the distance from node p to node q and then to x is d. under such condition, e_q=d,c_q=k (for all the j, $v_{q,p}(h_j(x))$=d). Now, another file y is put into the Internet and the corresponding dwBF is adjusted. Suppose one of the slots used by x is used by y too. If y is placed at a distance farer than x, and $v_{q,p}(h_j(x))$ remains to be d. On the contrary, If y is placed at a distance closer than x, and $v_{q,p}(h_j(x))$ is the distance to y, which can be shorter than d. So, dwBF evaluates a longest distance, $v_{q,p}(h_j(x))$, named e_q. Given that it gives a smaller scale of the actual distance, the slot e_q provided is likely to be shared by another file z. However, if the value of many h_j (x) slot is e_q, it is very likely to be the actual distance to x. To take all these into consideration, c_q is to be calculated.

If all the bits of h_i(x) are overlaid by a set of files at the place of x or near x, a false direction may happen in the router process. However, even if several nodes are under such condition, failure may happen at some subsequent nodes in the router path. If the files which overlay x are distributed in the whole Internet probabilistically, the distance to some overlaid files will be increased and the process may choose a better router. This will form a router correcting mechanism, yet there is still possibility of router failure.

To test the effectiveness of the routing algorithm, we conducted simulation experiments. It preliminary demonstrate the potential of our approach. We use Transit-Stub model of GT-ITM[11] to generate a network topology. The following figure 3 presents the results for 1008 node Transit-Stub networks. All network is static; i.e. no link or interface fails, no node joined or left.

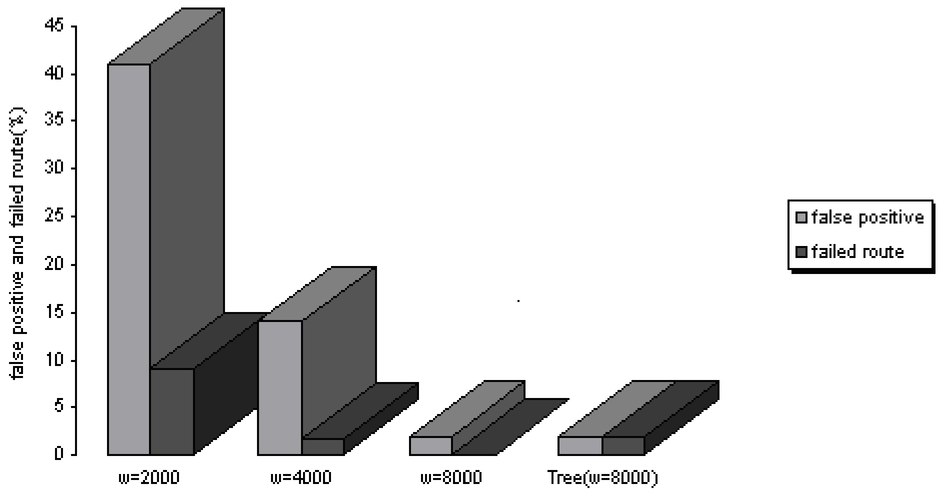

Figure 3: Results of simulation

According to the results, the route failure was 0.006%, which is lower than the expected rate (2%) of false positives in standard Bloom Filters.

3. Conclusion and Future Work

Actually, we can use Bloom filter in several ways. For example, some extra bits can be added in slots to control visit. In this way, the publishers can strengthen their control on the resources, which are effective in nominating and preventing pirating. It is easy to deal with replication. For each replication, dwBF can be set in accordance with the ways discussed above. Router process chooses a nearer replication.

Also, we can put other notions of expense, time and other evaluations into Bloom filter instead of distance.

The storage for dwBF on node p is $o(n|P_p|)$ while rate of false positives is fixed. n is the number of distinct files in the network; $|P|$ is the number of peers of the node. Since it is linear in n, the storage requirement is quite large. For 1M nodes with 100 files in each, the storage requirement of each node is about 10GB. That means it does not scale well. How to overcome this scalability problem is also our future work.

4. References

[1] B. Bloom. Space/time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*. 1970, **13**(7):422-426.

[2] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. *Proceedings of ACM SIGCOMM*. San Diego, California, USA. August 2001. ACM Press, 149-160

[3] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-scale Peer-to-peer System. *IFIP/IEEE International Conferenceo on Distributed Systems Platforms (Middleware)*. 2001, pp.329-350.

[4] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz. Tapestry: A Resilient Globalscale Overlay for Service Deployment. *IEEE Journal on Selected Areas in Communications*. 2004, **22**(1):41-53.

[5] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The Design and Implementation of an International Naming System. *Proc. of 17th ACM Symposium on Operating System Principles*, 1999, pp. 186-201.

[6] M. Gritter and D. R. Cheriton. An Architecture for Content Routing Support in the Internet. *Proc. of USENIX USITS*. 2001, pp.37-48.

[7] S. E Czerwinski, B. Y. Zhao, T. D. Hodes, A. D. Joseph. and R. H. Katz. An Architecture for a Secure Service Discovery Service. *Proc. of MobiCom-99*. 1999, pp.24-35.

[8] P. H. Hsiao. Geographical Region Summary Service for Geographical Routing. *Proc. of ACM International Symposium on Mobile ad hoc Networking and Computing*. 2001, pp263-266.

[9] Sean C. Rhea and John Kubiatowicz. Probabilistic Location and Routing. *Proc. of INFOCOMM*. 2002, pp. 1248-1257.

[10] A. Broder and M. Mitzenmacher. Network Application of Bloom Filters: A Survey. *Internet Mathematics*. 2004, **1**(4):485-509.

[11] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to Model an Internetwork. *Proc. of INFOCOMM*. 1996, pp. 594-602.