A TOKEN-BASED BROADCAST ALGORITHM OVER DHT FOR LARGE-SCALE COMPUTING INFRASTRUCTURES

Kun Huang[1] and Dafang Zhang[2]
[1]*School of Computer and Communication,* [2]*School of Software, Hunan University*
Changsha, Hunan Province 410082, P. R. China

Keywords: Computing infrastructure, Peer-to-Peer network, Distributed hash table, Broadcast, Load balancing.

Abstract: Scalable and efficient broadcast is essential to the large-scale computing infrastructures such as PlanetLab and Grids. By exploiting the greedy routing mechanisms of Distributed Hash Table (DHT), existing DHT-based broadcast algorithms suffer from the limitations of scalability and load balancing, incurring high construction and maintenance overhead for a distributed broadcast tree (DBT). This paper presents a token-based broadcast algorithm over DHT for the large-scale computing infrastructures, where each node selects the finger nodes as its children by a token value in a top-down approach. Theoretical analysis and experimental results show that the token-based broadcast algorithm can construct and maintain a balanced DBT with low overhead, where the branching factors of each node are at most two, and the tree height is $O(\log n)$ in a Chord of n nodes, without any extra storage space and explicit maintenance overhead.

1 INTRODUCTION

The large-scale computing infrastructures such as PlanetLab (Bavier, et al., 2004) and Grids (Foster, et al., 2001) have become increasingly important for developing and evaluating many emerging distributed applications. Typically, these computing infrastructures consist of large numbers of personal workstations and dedicated servers scattered around the world. For example, PlanetLab (2008) currently consists of 870 nodes at 460 sites. As the size of the computing infrastructures continues to grow, it is very challenging to efficiently manage such large-scale dynamic distributed systems.

Distributed broadcast is one of the fundamental primitive operations of distributed information management systems (Renesse, et al. 2003; Yalagandula and Dahlin, 2004; Oppenheimer, et al. 2008; Jain, et al., 2007) to disseminate information on a global scale. For example, in PlanetLab, researchers replicate their programs, along with the corresponding execution environment, on tens of thousands of nodes before launching a distributed application. In recent years, the Peer-to-Peer (P2P) based broadcast algorithms have been proposed to perform scalable content distribution across the large-scale computing infrastructures. There are two design principles for a P2P-based broadcast algorithm according to overlay structures: tree-based and mesh-based approaches. The tree-based approach constructs a tree overlay rooted at the source node as the content delivering structure, such as ESM (Chu, et al., 2002), NICE (Baerjee, et al., 2002), Scribe (Castro, et al., 2002), and Bayeux (Zhuang, et al., 2001). Since the single tree structure is vulnerable to the failure of an interior node, the multiple-tree approaches such as SplitStream (Castro, et al., 2003) and CoopNet (Padmanbhan, 2002) are proposed to improve the resilience, where each sub-stream of content is delivered along one of multiple disjoint trees. The mesh-based approach such as Bullet (Kostic, et al., 2003), FastReplica (Cherkasova and Lee, 2003), Bullet' (Kostic, et al., 2005), BitTorrent (Cohen, 2003), and CoBlitz (Park, et al., 2006) constructs a data-driven mesh overlay as a swarm system, where each node has a small set of neighbours to exchange data. Despite of these arguments (Venkataraman, et al, 2006; Wang, et al, 2007) against the two approaches, the tree-based approach is more suitable for the large-scale computing infrastructures with a large fraction of relatively stable dedicated nodes due to its simplicity and controlled overhead.

It is crucial for a DHT-based broadcast algorithm to meet scalability, robustness and load balancing. First, the algorithm should scale well to a large

Huang K. and Zhang D. (2009).
A TOKEN-BASED BROADCAST ALGORITHM OVER DHT FOR LARGE-SCALE COMPUTING INFRASTRUCTURES.
In *Proceedings of the International Conference on Wireless Information Networks and Systems*, pages 123-130
DOI: 10.5220/0002200101230130

number of participating nodes with only a limited number of messages passing with respect to the size of these computing infrastructures. Also it should have low construction and maintenance overhead for a distributed broadcast tree (DBT). Second, the algorithm should be robust to the dynamics of node arrivals and departures. Finally, the algorithm should ensure good load balancing in the sense that the broadcast workload is evenly distributed among all the participating nodes in the DBT, without any performance bottleneck or hotspot of content delivery. Therefore, load balancing is essential for scalability and robustness of a P2P-based broadcast algorithm.

Recently, most DHT-based broadcast algorithms (Yalagandula, et al., 2004; Oppenheimer, et al., 2008; Jain, et al., 2007; Cai and Hwang, 2007; Castro, et al., 2002; Castro, et al., 2003; Padmanbhan, et al., 2002; Park and Pai, 2006; Zhuang, et al., 2001; Zhang, et al., 2003; Renesse and Bozdog, 2004; El-Ansary, et al., 2003; Ratnasamy, et al., 2001) have been proposed. These broadcast algorithms utilize the routing mechanisms of DHT to build up an overlay routing path between a source node and each destination node, and then construct a DBT rooted at the source node by merging all the routing paths in a top-down approach (El-Ansary, et al., 2003) or in a bottom-up approach (Castro, et al., 2002). However, such broadcast algorithms suffer from the limitations of scalability and load balancing. First, DHT is a greedy routing algorithm, where each node always forwards a searched key to the closest preceding node in its finger table, whose identifier is closer to the key in the identifier space. The greedy essence of DHT results in existing DHT-based broadcast algorithms construct a flat and unbalanced DBT either in a top-down approach (El-Ansary, et al., 2003) or in a bottom-up approach (Castro, et al., 2002). Second, existing DHT-based broadcast algorithms have high construction and maintenance overhead for a DBT with respect to a large number of participating nodes. For example, the reverse-path forwarding based broadcast algorithm (Castro, et al., 2002) requires interior nodes to contain its children for reverse forwarding using extra storage space, which leads to high overhead of maintaining these children. Moreover, although existing DHT-based broadcast algorithms adopt the pushdown and anycast methods (Castro, et al., 2002; Castro, et al., 2003) to tackle the overloading of nodes by adjusting the branching factors between nodes, Bharambe et al. (2005) indicate that these adjustment methods result in a significant number of non-DHT links that are present in the DBT but are not part of the routing links of DHT, which not only restricts the scalability of DBT but also incurs higher maintenance overhead of these non-DHT links due to the dynamic nodes.

To address the above issues, this paper presents a token-based broadcast algorithm over DHT for scalability and load balancing in the large-scale computing infrastructures. The key idea of the token-based broadcast algorithm is that by leveraging the topology and routing mechanisms of Chord, each node selects the finger nodes as its children by a token value in a top-down approach. Theoretical analysis and experimental results show that the token-based broadcast algorithm can construct and maintain a scalable and balanced DBT, where the branching factors of each node are at most two, and the tree height is $O(\log n)$ in a Chord of n nodes, without any extra storage space for each node and explicit maintenance overhead for the parent-child membership in the DBT.

The rest of the paper is organized as follows. Section 2 overviews the Chord network. In Section 3, we present in details the token-based algorithm. Section 4 presents our experimental results. Related work is introduced in Section 5 and Section 6 concludes the paper.

2 CHORD OVERVIEW

The Chord network is modelled as an undirected graph $G=(V,E)$, where the vertex set V contains n nodes and E is the set of overlay links between nodes. According to the identifiers of nodes, Chord organizes nodes as a ring topology in the circular space. An object's identifier k is assigned to the first node whose identifier is equal to or follows k in the identifier space of Chord. This node is called the successor node of the identifier k, denoted by $Succ(k)$. In Chord, $Pred(u)$ refers to the immediate predecessor of a node u, while $Succ(u)$ refers to the immediate successor of a node u. Besides its immediate predecessor and successor, each node u also maintains a set of m finger nodes that are spaced exponentially in the identifier space of Chord. The i^{th} finger node of a node u, denoted by $Finger(u,i)$, is the first node that succeeds u by at least 2^i in the identifier space, that is $Finger(u,i)=Succ((u+2^i) \bmod 2^m)$, where $0 \le i \le m-1$.

Chord adopts a greedy finger routing algorithm (Stoica, et al., 2001) to recursively (or iteratively)

forward a query message with an object's identifier k to its successor node $Succ(k)$ that maintains a pair (k,v), where v is the object's value. When a node u want to lookup an object's identifier k, it forwards a query message with the identifier k to its finger node $Finger(u, j)$, which is closest to the successor node $Succ(k)$ in the circular identifier space, satisfying $Finger(u, j) \in (u, Succ(k)]$ and $Min\{Dist(Finger(u, j), k), 0 \le j \le m-1\}$, where $Dist(u_1, u_2)$ is the numeric distance between two identifiers u_1 and u_2, that is $Dist(u_1, u_2) = (u_1 - u_2 + 2^m) \bmod 2^m$. And then the finger node $Finger(u, j)$ continues to forward the query message to the next node using the similar routing algorithm, until to the successor node $Succ(k)$. Therefore, the finger routing algorithm of Chord is a scalable and efficient lookup algorithm, with average routing path length of $O(\log n)$ and average node state space of $O(\log n)$ in a Chord of n nodes.

3 TOKEN-BASED BROADCAST

3.1 Token-based Broadcast Algorithm

This section presents a token-based broadcast algorithm over DHT to achieve the scalability and load balancing. In the token broadcast algorithm, each node adaptively selects the finger nodes as its children by a token value, and then a balanced DBT is constructed in a top-down approach.

The DBT construction process of the token-based broadcast algorithm is as follows. First, on receiving a broadcast message with a token value t, each node N_i selects the t^{th} finger node $Finger(N_i, t)$ as its left child and the $(t+1)^{th}$ finger node $Finger(N_i, t+1)$ as its right child. Second, N_i forwards the broadcast message with the incremental token value $t+1$ to its children, until no any child is selected. Note that when receiving a broadcast message with the token value t, each node N_d selects its left child N_l and right child N_r, and then checks to see whether its children's identifiers are not beyond the identifier of the source node, satisfying $Dist(N_d, N_l) < Dist(N_d, N_s)$ and $Dist(N_d, N_r) < Dist(N_d, N_s)$. If not, N_d does not forward the broadcast message to N_l or N_r. When initiating a broadcast message with a token value $t = 0$, a source node N_s must adjust the token value t to skip its duplicated finger nodes, by checking whether the condition $Finger(N_s, t) \neq Finger(N_s, t+1)$ is satisfied. If not, then the token value is incremented as $t+=1$ and N_s continues to check the above condition. Figure 1 illustrates the pseudo-codes of the token-based broadcast algorithm.

```
1:  // receive (P, R, Token) represents that node P receives
      a broadcast message from source node R with Token.
      The initial value of Token is 0.
2:  // Skip redundant fingers by adjustting token value
3:  if Token≥m then
4:    exit (0);
5:  else
6:    for i←Token to m-2 do
7:      if Finger(P, i)≠Finger(P, i+1) then
8:        break;
9:      else
10:       Token←i+1;
11:     endif
12:   endfor
13: endif
14: // Select left child for broadcasting
15: if Finger(P, Token) ∈ (P, R) then
16:   Left←Finger(P, Token);
17:   send (Left, R, Token+1);
18: endif
19: // Select right child for broadcasting
20: if Token < m then
21:   if Finger(P, Token+1)∈ (P, R) then
22:     Right←Finger(P, Token+1);
23:     send (Right, R, Token+1);
24:   endif
25: endif
```

Figure 1: Pseudo-codes of token-based broadcast algorithm.

Figure 2 depicts a balanced DBT construction example using the token-based broadcast algorithm in a 16-node Chord, where there is the uniform identifier space. In Figure 2(a), the source node N_0 initiates a broadcast message with a token value $t = 0$, and selects $N_1 = Finger(N_0, 0)$ and $N_2 = Finger(N_0, 1)$ as its left and right children respectively, and then forwards the broadcast message with the incremental token value $t = 1$ to N_1 and N_2; N_1 and N_2 continues to select their children and increment the token value, and then further forward the broadcast message, until no any child is selected. When receiving the broadcast message with the token value $k = 3$, N_8 to N_{14} have no any child and don't forward any broadcast message, since the identifiers of their children are more than the identifier of the source node N_0. Figure 2(b) illustrates that the corresponding balanced DBT is constructed from the finger routing paths in Figure 2(a). Therefore, Figure 2 shows that

the token-based broadcast algorithm constructs a balanced DBT, where the branching factors of each node are at most two, and the tree height is $\lceil \log_2 n \rceil = 4$, in which $n = 16$ is the number of nodes in Chord.

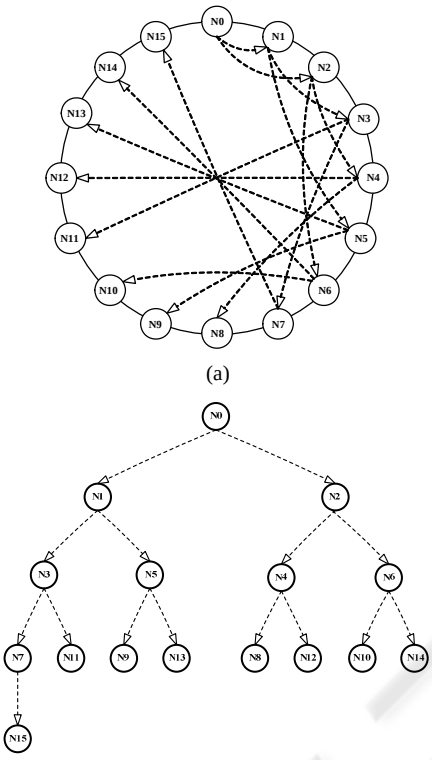


Figure 2: Balanced DBT construction using token-based broadcast algorithm in a 16-node Chord. (a) Finger routing paths rooted at N_0. (b) Constructed balanced DBT rooted at N_0.

3.2 Algorithm Analysis

In essence, the token-based broadcast algorithm is to select the appropriate children for each node using a token value to alternately cover all destination nodes. Unlike the k-ary search based broadcast algorithm, the token-based broadcast algorithm adopts the exponential growth of the interval distance to forward a broadcast message. So it is guaranteed by the novel children selection strategy that the token-based algorithm can construct a balanced DBT and eliminate the duplicated broadcast messages between nodes.

Figure 3 illustrates the properties of token-based broadcast algorithm in a 16-node Chord. Figure 3(a) depicts the case that a source node N_0 forwards a broadcast message to a destination node N_{15}. The

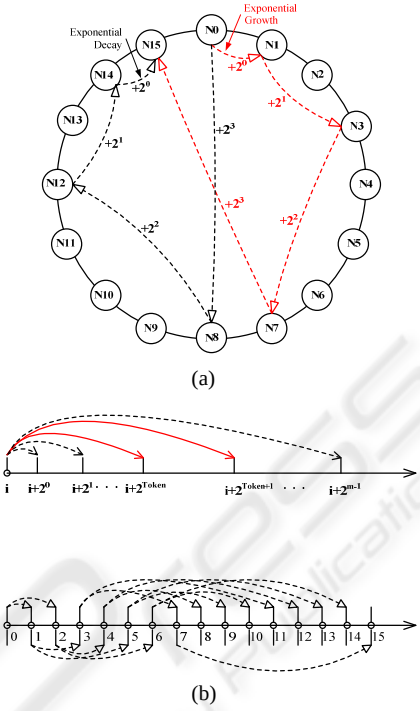


Figure 3: Properties of token-based broadcast algorithm in a 16-node Chord. (a) Finger routing paths rooted at N_0 with different exponential interval distance. (b) Alternately covering all the destination nodes to construct balanced DBT rooted at N_0.

token-based algorithm selects the children with the exponential growth of the interval distance such as 2^0, 2^1, 2^2, and 2^3, whereas the k-ary search based algorithm selects the children with the exponential decay of the interval distance such as 2^3, 2^2, 2^1, and 2^0. As seen in Figure 3(b), when a source node N_0 forwards a broadcast message to all destination nodes, each node i selects $Token^{th}$ and $(Token+1)^{th}$ finger nodes as its left and right children respectively. The interval distance between parent and children in such circular space is exponentially growing and all destination nodes $N_1, N_2 \cdots N_{15}$ are alternately covered to construct a balanced DBT root at N_0.

Also, the DBT constructed by the token-based broadcast algorithm is scalable, where the tree height is $O(\log n)$ in which n is the number of nodes in Chord. We assume that a source node is u and a destination node is v, satisfying $u \neq v$. Since there is the uniform identifier space of Chord, u adjusts the token value $Token = t$ to skip the duplicated finger nodes in its finger table, and then forwards the incremental token value until it is $Token = m$. Thus,

the finger routing path from u to v is $u+2^t \to u+2^t+2^{t+1} \to \cdots \to v$, which has $h=m-t$ hops, and we compute $Max(Dist(u,v))=2^m-2^t$ in which 2^t is the average range between two immediately adjacent nodes in Chord. Due to $2^t=2^m/n$, we deduce $h=m-t=\log_2 n$. Hence, it is proved that the tree height is $O(\log n)$ and the DBT is scalable.

Moreover, the token-based broadcast algorithm needs no explicit construction and maintenance overhead for such balanced DBT. First, since the token-based algorithm selects the children only by a token value in a top-down approach, each parent does not need extra storage space for its children. So the balanced DBT is implicitly constructed from the finger routing paths of Chord. Second, on node arrivals and departures, Chord adopts a finger stabilization algorithm (Stoica et al., 2001) to periodically update the finger tables of nodes. But the token-based algorithm does not need extra cost to repair the parent-child membership, which significantly reduces the maintenance overhead of the balanced DBT. Therefore, the DBT constructed by the token-based broadcast algorithm has no any extra storage cost and explicit maintenance overhead of each node.

4 EVALUATION

This section presents the simulation experiments to validate the token-based broadcast algorithm. We implement three DHT-based broadcast algorithms including the k-ary search based broadcast algorithm, the reverse-path forwarding based broadcast algorithm, and the token-based broadcast algorithm.

The performance metrics of a DHT-based broadcast algorithm include branching factor, path length, message overhead, and message imbalance factor. Branching factor refers to the number of a node's children in a DBT. Path length refers to hops of a finger routing path from a source node to a destination node. Message overhead refers to the number of messages for constructing and maintaining a DBT. Message imbalance factor refers to the ratio between the maximal number of messages and average number of messages on each node in a DBT. In the experiments, we simulate from 16 to 2048 Chord nodes to examine the performance metrics of above three algorithms.

Figure 4 depicts the comparison of branching factor. Figure 4(a) shows that there is the same maximal branching factor of $\log_2 n$ for both the k-ary search based and reverse-path forwarding based algorithms, whereas there is the constant maximal branching factor of 2 for the token-based algorithm. For example, in a 2048-node Chord, both the k-ary search based and reverse-path forwarding based algorithms have the same maximal branching factor of 11, while the token-based algorithm has the maximal branching factor of 2. As seen in Figure 4(b), both the k-ary search based and reverse-path forwarding based algorithms construct a skewed DBT, while the token-based algorithm constructs a balanced DBT. For example, about 88% of nodes in the DBT constructed by both the k-ary search based and reverse-path forwarding based algorithms has the branching factors of equal to or less than 2 and the remaining 12% have the branching factors from 3 to 11. In the token-based algorithm, about 50% of nodes in the DBT have the branching factors of 2, and the remaining 50% are almost leaf nodes without any branching factor. Therefore, the token-based broadcast algorithm constructs a balanced DBT, where the branching factors of each node are at most two.

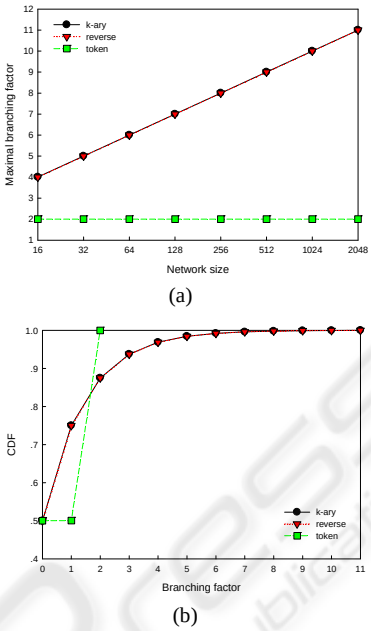
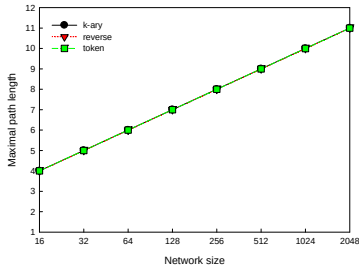


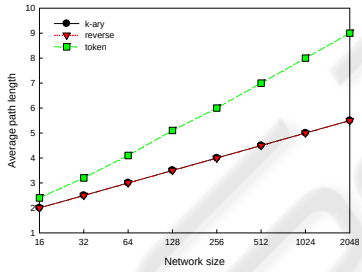
Figure 4: Branching factor. (a) Maximal branching factor. (b) Cumulative distribution functions in a 2048-node Chord.

Figure 5 depicts the comparison of path length. Figure 5(a) shows that the token-based algorithm has the same maximal path length of $\log_2 n$ with both the k-ary search based and reverse-path forwarding based algorithms. As seen in Figure 5(b), compared to both the k-ary search based and reverse-path

forwarding based algorithms, the token-based algorithm increases 20%~40% of average path length. The root cause lies in the fact that both the k-ary search based and reverse-path forwarding based algorithms construct a skewed and fat DBT from the shortest finger routing paths, whereas the token-based algorithm constructs a balanced and narrow DBT from the novel finger routing paths. Despite that, a DBT constructed by the token-based algorithm guarantees the same max broadcast latency of $\log_2 n$ with both the k-ary search based and reverse-path forwarding based algorithms. Therefore, the token-based broadcast algorithm construct a scalable DBT, where the tree height is $O(\log n)$ in which n is the number of nodes in Chord.



(a)



(b)

Figure 5: Path length. (a) Maximal path length. (b) Average path length.

Figure 6 depicts the comparison of message overhead in the uniform identifier space of Chord. As seen in Figure 6, both the k-ary search based and token-based algorithms have the same messages of $n-1$, while the reverse-path forwarding based algorithm has $2(n-1)$ messages. The root cause lies in the fact that the reverse-path forwarding based algorithm constructs a DBT in a bottom-up approach and a broadcast message is forwarded along the reverse finger routing paths of Chord, while both the k-ary search based and our token-based algorithms construct a DBT in a top-down approach and a broadcast message is forwarded along the finger routing paths. When a node arrives and departs, the finger stabilization algorithm (Stoica et al., 2001) of Chord produces the average messages of $O(\log^2 n)$ for maintaining the finger tables of relative nodes. Thus the reverse-path forwarding based algorithm needs extra average messages of $O(\log^2 n)$ to maintain the parent-child membership of each node in the DBT. Therefore, the token-based broadcast algorithm constructs and maintains a balanced DBT without any extra storage cost for the children of each node and explicit maintenance overhead for the parent-child membership in the DBT.

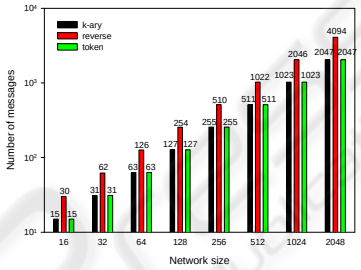


Figure 6: Message overhead.

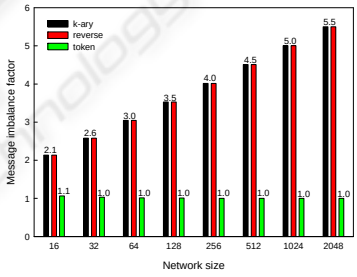


Figure 7: Message imbalance factor.

Figure 7 depicts the comparison of message imbalance factor. As seen in Figure 7, both the k-ary search based and reverse-path forwarding based algorithms have the same message imbalance factor as a function of network sizes, while the token-based algorithm has the constant message imbalance factor independent of network sizes. For example, in a 2048-node Chord, the message imbalance factor is about 5.5 for both the k-ary search based and reverse-path forwarding based algorithms, while the message imbalance factor is kept about 1.0 for our token-based algorithm. Compared to both the k-ary search based and reverse-path forwarding based algorithms, the token-based broadcast algorithm reduces 50%~82% of the message imbalance factor, which further validates that a DBT constructed by the token-based algorithm is balanced.

5 RELATED WORK

There have been many research efforts on multicast and broadcast in past decades. In general, broadcast is a special case of multicast. In recent years, application-level multicast and broadcast algorithms are categorized into tree-based and mesh-based approaches as follows.

In the tree-based approach, a tree overlay rooted at a source node is constructed as the content delivering structure. For example, ESM (Chu, et al., 2002) is a classic application-layer multicast system towards small-sized groups. NICE (Baerjee, et al., 2002) is proposed to support a larger number of participating nodes using a hierarchical clustering approach. Bayeux (Zhuang, et al., 2001) is proposed to construct a scalable and fault-tolerant application-layer multicast system by replicating root nodes and clustering receivers via identifier on top of Tapestry (Zhao, et al., 2002). Since the single tree structure is vulnerable to node dynamics, the multiple-tree approach has been proposed, where a forest with multiple disjoint sub-trees is constructed and each sub-stream of the content is delivered along each sub-tree. For example, SplitStream (Castro, et al., 2003) is proposed to construct an interior-node disjoint forest of multiple Scribe (Castro, et al., 2002) trees on top of Pastry (Rowstron and Druschel, 2001). CoopNet (Padmanbhan, et al., 2002) is proposed to compute locally random or node-disjoint forests of trees, primarily designed for resilience to node departures.

In the mesh-based approach, a data-driven mesh overlay is constructed as a swarm system, where each node has a small of neighbours to exchange data. For example, Bullet (Kostic, et al., 2003) and Bullet' (Kostic, et al., 2005) are proposed for large-file distribution in the wide area, where a overlay mesh is constructed over any overlay tree and each node transmits a disjoint set of data to its children in order to maintain uniform distribution of each data and achieve high throughput. BitTorrent (Cohen, 2003; Huang, et al., 2008) is one of the most popular P2P content distribution systems, where all participating nodes upload and download equal-sized blocks in parallel. These mesh-based swarm systems can mitigate link stresses and the performance bottleneck of the origin, but can incur enormous traffic stresses on the Internet Service Providers (ISP).

Moreover, distributed aggregation is also one of the fundamental primitive operations. Aggregation is to recursively compute the global information by applying an aggregate function such as min, max, count, and sum on a set of local status in the large-scale computing infrastructures. DHT-based aggregation algorithms for global resource monitor and discovery related to our work have been recently proposed. For example, Astrolabe (Renesse, et al., 2003) is a DNS-like distributed management service by organizing the resources into a hierarchy of domains, specifying an aggregation tree between domains, and exchanging information across domains using an unstructured gossip protocol. SDIMS (Yalagandula and Dahlin, 2004) is a scalable distribution information system, where each attribute is hashed to a key and the aggregation tree rooted at the key is built upon the routing mechanisms of Plaxton. Other algorithms also include SOMO (Zhang, et al., 2003), Willow (Renesse and Bozdog, 2004), and DAT (Cai and Hwang, 2007). The token-based broadcast algorithms combined with these DHT-based aggregation algorithms can provide a scalable and load-balanced distributed information management in the large-scale computing infrastructures.

6 CONCLUSIONS

It is critical for the large-scale computing infrastructures to support for scalable and efficient broadcast. Existing DHT-based broadcast algorithms suffer from the limitations of scalability and load balancing. This paper presents a token-based broadcast algorithm over DHT for the large-scale computing infrastructures, where by leveraging the topology and routing mechanisms of Chord, each node selects the finger nodes as its children by a token value in a top-down approach. Experimental results show that compared to existing broadcast algorithms, the token-based broadcast algorithm reduces 50%~82% of the message imbalance factor. Therefore, the token-based broadcast algorithm constructs a scalable and balanced DBT, without any extra storage cost and explicit maintenance overhead, which is suitable for the large-scale computing infrastructures.

ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation of China under grant No.60673155 and No.90718008.

REFERENCES

Bavier, A., et al., 2004. Operating system support for planetary-scale network services. In: 1st Symposium on Networked Systems Design and Implementation, pp. 253-266.

Foster, I., Kesselman, C., Tuecke, S., 2001. The anatomy of the grid: enabling scalable virtual organizations. International Journal of Supercomputer Applications, 15(3), pp. 200-222.

PlanetLab, 2008. Availabe at: http://www.planet-lab.org.

Renesse, R. V., Birman, K. P., Vogels, W., 2003. Astrolabe: a robust and scalable technology for distributed system monitoring, management, and data mining. ACM Transaction on Computer Systems, 21(2), pp. 164-206.

Yalagandula, P., Dahlin, M., 2004. A scalable distributed information management system. In: ACM SIGCOMM, pp. 379-390.

Oppenheimer, D., Albrecht, J., Patterson, D., Vahdat, A., 2008. Design and implementation tradeoffs for wide-area resource discovery. ACM Transactions on Internet Technology, 8(2), pp.1-40.

Jain, N., et al., 2007. STAR: self-tuning aggregation for scalable monitoring. In: Proc. of VLDB, pp. 962-973.

Cai, M, Hwang, K., 2007. Distributed aggregation algorithms with load-balancing for scalable grid resource monitoring. In: 21st International Parallel and Distributed Processing Symposium, Long Beach, California, USA.

Castro, M., Druschel, P., Kermarrec, A.-M., Rowstron, A., 2002. SCRIBE: a large-scale and decentralized application-level multicast infrastructure. IEEE Journal on Selected Areas in Communication, 20(8), pp. 1489-1499.

Castro M., et al., 2003. Splitstream: high-bandwidth multicast in cooperative environments. In: 19th ACM Symposium on Operating Systems Principles, pp. 298-313.

Padmanbhan, V. N., Wang, H. J., Chou, P. A., Sripanid-Kuchai, K., 2002. Distributed streaming media content using cooperative networking. In: ACM NOSSDAV, pp. 177-186.

Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A., 2003. Bullet: high bandwidth data dissemination using an overlay mesh. In: 19th ACM Symposium on Operating Systems Principles, pp. 282-297.

Cherkasova, L. Lee J., 2003. FastReplica: efficient large file distribution within content delivery networks. In: USENIX Symposium on Internet Technologies and Systems, Seattle, Washington, USA.

Kostic, D., et al., 2005. Maintaining high bandwidth under dynamic network conditions. In: USENIX Annual Technical Conference, pp. 193-208.

Ganguly, S, Saxena, A, Bhatnagar, S, Banerjee, S., 2005. Fast replication in content distribution overlays. In: IEEE INFOCOM, pp. 2246-2256.

Park, K., Pai, V. S., 2006. Scale and performance in the CoBlitz large-file distribution service. In: 3rd Symposium on Networked Systems Design and Implementation, pp. 29-44.

Chu, Y., Rao, S. G., Seshan, S., Zhang, H., 2002. A case for end system multicast. IEEE Journal on Selected Areas in Communication, Special Issue on Networking Support for Multicast, 20(8), pp. 1456-1471.

Baerjee, S., Bhattacharjee, B., Kommareddy, C., 2002. Scalable application layer multicast. In: ACM SIGCOMM, pp. 205-217.

Zhuang, S., et al., 2001. Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination. In: ACM NOSSDAV, pp. 11-20.

Cohen, B., 2003. Incentives build robustness in BitTorrent. In: Workshop on Economics of Peer-to-Peer Systems, Berkeley, California, USA.

Venkataraman, V., Yoshida, K., Fancis, P., 2006. Chunkspread: heterogeneous unstructured end system multicast. In: IEEE ICNP, pp. 2-11.

Wang, F., Xiong, Y., Liu, J., 2007. mTreebone: a hybrid tree/mesh overlay for application-layer live video multicast. In: IEEE ICDCS, Toronto, Ontario, Canada.

Stoica, I., et al., 2001. Chord: a scalable peer-to-peer lookup service for Internet applications. In: ACM SIGCOMM, pp. 149-160.

Ratnasamy, S., et al., 2001. A scalable content addressable network. In: ACM SIGCOMM, pp. 161-172.

Rowstron, A. Druschel, P., 2001. Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Lecture Notes in Computer Science, 2218, pp. 329-351.

Zhao, B., Kubiatowicz, J., Joseph, A., 2002. Tapestry: a fault-tolerant wide-area application infrastructure. ACM Computer Communication Review, 32(1), pp. 81-81.

Zhang, Z., Shi, S.-M., Zhu, J., 2003. SOMO: self-organized metadata overlay for resource management in P2P DHT. In: International Workshop on Peer-to-Peer Systems.

Renesse, R. V., Bozdog, A., 2004. Willow: DHT, aggregation, and publish/subscribe in one protocol. In: International Workshop on Peer-to-Peer Systems.

El-Ansary, S., Alima, L.O., Brand, P., Haridi, S., 2003. Efficient broadcast in structured P2P networks. In: International Workshop on Peer-to-Peer Systems.

Ratnasamy, S., Handley, M., Karp, R., Shenker, S., 2001. Application-level multicast using content-addressable networks. In: International Workshop on Networked Group Communicaion, pp. 14-29.

Bharambe, A. R., et al., 2005. The impact of heterogeneous bandwidth constraints on DHT-based multicast protocols. In: International Workshop on Peer-to-Peer Systems, 2005.

Huang, K., Wang, L., Zhang, D., Liu, Y., 2008. Optimizing the BitTorrent performance using adaptive peer selection strategy. Future Generation Computer Systems, 24(7), pp. 621-630.