IEICE TRANS. INF. & SYST., VOL.E90–D, NO.9 SEPTEMBER 2007

LETTER

Reducing Replication Overhead for Data Durability in DHT Based P2P System

Kyungbaek KIM[†a)], ***Student Member and*** **Daeyeon PARK**[†], ***Member***

SUMMARY DHT based p2p systems appear to provide scalable storage services with idle resource from many unreliable clients. If a DHT is used in storage intensive applications where data loss must be minimized, quick replication is especially important to replace lost redundancy on other nodes in reaction to failures. To achieve this easily, a simple replication method directly uses a consistent set, such as a leaf set and a successor list. However, this set is tightly coupled to the current state of nodes and the traffic needed to support this replication can be high and bursty under churn. This paper explores efficient replication methods that only glimpse a consistent set to select a new replica. Replicas are loosely coupled to a consistent set and we can eliminate the compulsory replication under churn. Because of a complication of the new replication methods, the careful data management is needed under churn for the correct and efficient data lookup. Results from a simulation study suggest that our methods can reduce network traffic enormously for high data durability.

key words: *peer-to-peer, DHT, replication, data durability*

1. Introduction

In these days, peer-to-peer systems have become an extremely popular platform for large-scale content sharing, even the p2p based file systems appear. Unlike client/server model based storage systems, which centralize the management of data in a few highly reliable servers, peer-to-peer storage systems distribute the burden of data storage and communications among tens of thousands of clients. The wide-spread attraction of this model arises from the promise that idle resources may be efficiently harvested to provide scalable storage services. To promise that, a lot of research papers discussed the Distributed Hash Table (DHT) based p2p routing algorithms [1]–[3] and the p2p system which uses the DHT based p2p routing algorithm is called the structured p2p system.

Many structured p2p systems use a simple replication method to cope with massive node failures [4], [5], [7]. Figure 1 (a) shows a simple replication. In this figure, we assume that the object range of node A is from NodeID C to NodeID A and node A stores objects whose ObjectID is on its object range. If a p2p system needs N replicas to achieve acceptable data durability, each node has N or more replicas for its object range on its sequentially closest neighbor nodes. This simple approach exploits a consistent set which is composed of M nodes with numerically closest nodeIDs relative to a present node's nodeID, such as a successor list of Chord [1] and a leaf set of PASTRY [2]. Basically, this set is used to conserve the routing information to cope with node failures. When a node joins or leaves, consistent sets of its neighbor nodes which detect change of membership should be update automatically in order to preserve the current state of the p2p system. That is, a consistent set is tightly coupled to the current network state. When node B leaves, node A recognizes this change by the automated update of its consistent set and selects the closest and unselected node of its consistent set, node F, for a new replica. Moreover, when a node (node B) leaves, its neighbor node (node D) automatically services the data of the object range of the leaving node immediately, because neighbor nodes of the leaving node already have the replicas of its stored data.

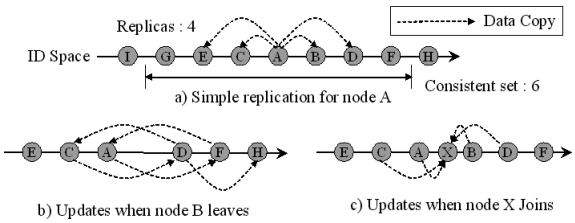


Fig. 1 Simple replication in DHT based p2p.

This simple replication which keeps replicas on the sequentially closest neighbor nodes guarantees the simple data durability management and the simple lookup under churn easily and automatically. However, this sequential replication is tightly coupled to the consistent set and when membership of nodes change, the replicas should be updated regardless of data durability. This compulsory copy takes high and bursty traffic under churn. In Fig. 1 (b), when node B leaves, nodes A, C, D and F which already have replicas on node B should make new replicas. Moreover, node D which is newly responsible for the object range of node B makes the replica for this range on node H. When a node joins, it changes the network state and each closest neighbor nodes of the new node update their consistent sets. Each affected neighbor node copies data of its object range to the new node as a new replica like Fig. 1 (c). In this case, when node B joins and leaves very frequently, compulsory data replications occur and heavy data traffic waste even if the dynamic behavior of node B can not affect data durability.

In our paper, we suggest the efficient replication methods to achieve the high data durability with small mainte-

Manuscript received July 21, 2006.
Manuscript revised April 2, 2007.

[†]The authors are with the Department of Electrical Engineering and Computer Science[Division of Electrical Engineering], Korea Advanced Institute of Science and Technology, 373–1 Guseong-dong Yuseong-gu, Daejeon, 305–701, Korea.
a) E-mail: kbkim@sslab.kaist.ac.kr

DOI: 10.1093/ietisy/e90–d.9.1452

nance cost. To reduce the compulsory copies which occur under churn, we interleave replicas on a consistent set and replicas are loosely coupled to a consistent set which is tightly coupled to the current network state. We propose *Quorum based replication* and *Availability based replication*. In the way of getting data durability, these two methods are different from each other. In Quorum based replication, it assumes that each node has a mean node availability of 0.5 based upon the measurements in [6] and each node keeps the number of replicas more than the target quorum to achieve target data durability. This method selects a new replica randomly among the consistent set. However, Availability based replication predicts node availability and select more reliable nodes as replicas to delay replication timing and to reduce network cost. This method guarantees the high data durability by a numerical value.

2. Proposed Idea

2.1 Quorum Based Replication

To prevent the compulsory data copy, we propose the quorum based replication. In this replication, we choose a mean node availability of 0.5 and get data durability by the number of replicas like the following equations. In these equations, D means data durability and a_n means node availability. The dominator of data durability is q which is the number of replicas and we set the quorum to achieve the high data durability.

$$\begin{aligned} D &= 1 - P(All\ replicas\ fail) \\ &= 1 - (1 - a_n)^q \end{aligned}$$

Like the Fig. 2, we add new information; the replication set that indicates which node replicates data. The size of this set is same to the size of the consistent set. When a node joins/leaves, the affected consistent set should be updated for the correctness of the routing mechanism, but data copy only occurs when the number of replicas which is indicated by the replication set is fewer than the quorum. That is, the replication is loosely coupled to the consistent set and the replicas are interleaved on the consistent set like the Fig. 2.

When a node leaves, the affected nodes choose new replicas only if it has a replica. In Fig. 2, when node B leaves, node A only updates the replication set because node B has no replica. However, when node D leaves, node A should select a new replica to keep up the quorum. When a node needs a new replica, it selects the numerically closest node from this node, because a member node which is numerically farther from this node withdraws from the consistent set with higher probability under churn. In the Fig. 2, if node A needs one more replica, node B can be selected as a new replica rather than node E.

2.2 Availability Based Replication

The quorum based replication tries to keep the number replicas above the quorum. However, if a new replica is assigned

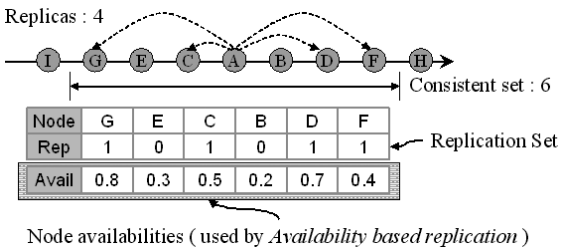


Fig. 2 Metadata for our replication methods.

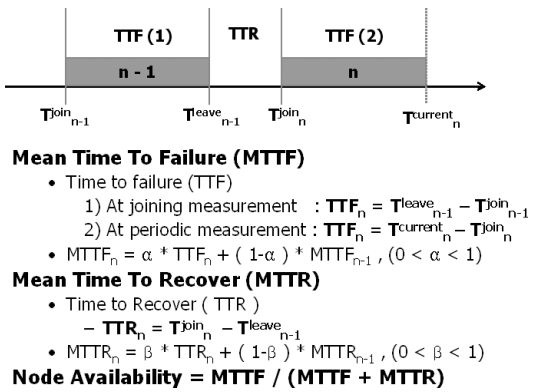


Fig. 3 Node availability prediction.

on a node which has the low node availability, this node may leave soon and another new replica is needed. If a new replica is chosen with a node availability, we can select the more available node as a replica and can reduce the overhead. To achieve this, we add the availability information of the replication set like Fig. 2. We get data durability like the following equations where $a_1, a_2, ..a_n$ is node availabilities of replicas. In this case, we set the target threshold of data durability as a numerical value.

$$D = 1 - (1 - a_1)(1 - a_2) \ldots (1 - a_n)$$

Like the quorum based replication, the availability based replication uses the replication set and the replicas are loosely coupled to the consistent set. The main difference of these replications is the selection mechanism for a new replica. In the availability based replication, each node computes the data durability with the node availabilities of replicas and the replication only occurs when the data durability is below the target threshold. When a new replica is needed, a node selects the most available node of the consistent set.

We assume that the node availability is the prediction value how long a node is alive after it joins the p2p system, because in other researches [6] the long lived nodes generally have the large bandwidth and the big computing power. Figure 3 shows this availability prediction mechanism. We use the Mean Time To Failure and the Mean Time To Recover to estimate the node availability. MTTF is the average value how long a node is alive after it joins and MTTR is the average value how long a node is sleep after it leaves. We can get TTF and TTR by using the last join time, the

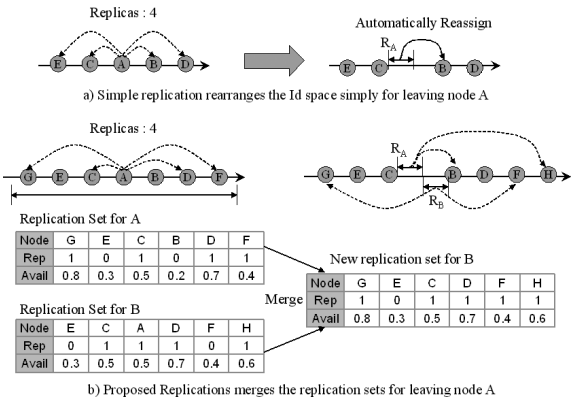


Fig. 4 Management for the replication set when node A leaves.

last leave time and the current join time. Unlike TTR, we periodically update the TTF by using the current time and the current join time. The MTTF and MTTR is obtained by the weighted average of TTF and TTR. In this case, α and β are set to 0.9. The node availability is computed with the equation, MTTF/(MTTF + MTTR).

Node availability is computed by each node and each node advertises it to all members of the consistent set by using the piggyback method. To detect a node failure, a node periodically sends a ping message to all member of the consistent set. Node availability is piggybacked on this ping message. According to this, each node maintains the node availabilities of the consistent set.

2.3 Management for the Replication Set

In the Fig. 4 (a), when node A leaves, the lookup request for data in node A is forwarded to node B automatically because of the basic concept of the DHT based p2p. In a simple replication, node B always has the replicas for node A and already has the data of the object range of leaving node A (R_A). That is, the rearrange of the ID space performs easily and automatically. However, in our replications, the replicas are interleaved on the consistent set and it can not be sure that node B has the replicas for node A. In order to copy the data of R_A, node B should know the locations of the replicas for node A. To do this, each node advertises its replication sets to all members of its consistent set with piggybacked ping message.

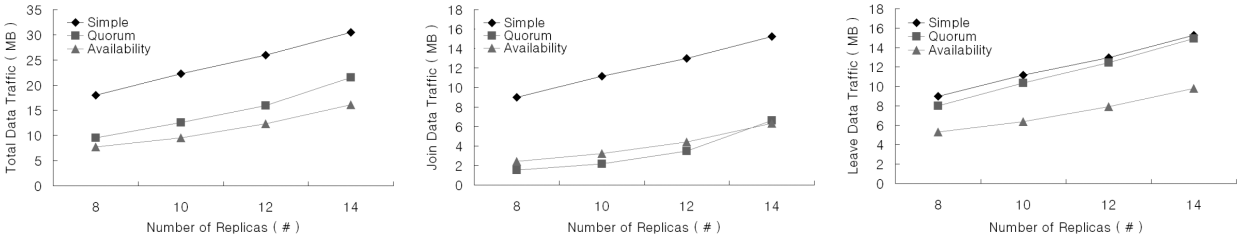
Figure 4 (b) shows the management of the replication set when node A leaves for our replication methods. The closest neighbor node B should be responsible for both of the object range of leaving node A (R_A) and the pervious object range of node B (R_B). There are two replication sets for R_A and R_B and they should be merged to one new replication set for node B. The basic rule of the merging operation is that if a node is a replica for any set, it remains a replica for the new set. In the Fig. 4 (b), node F is a replica for node A and it becomes a new replica for the new replication set. Otherwise, node E is not a replica for any set and it is still not used after merging. After merging operation, each replication method checks whether the new replication set needs new replicas or not.

During this merge operation, the real data are selectively copied to the replicas of the new replication set. The new responsible node B does not have the data of R_A and copies them from any one of previous replicas of node A. Because node H which is the previous replica of node B already has the data of R_B, the data of R_A are only copied to node H when the merging occurs. According to the same reason, node F and node G which are the previous replicas of node A only obtain the data of R_B during the merging operation. This selective copy can reduce the traffic overhead of the replication.

3. Evaluation

We make our p2p simulator which emulates behavior of nodes on the application layer. We implement a DHT based p2p algorithm, Pastry and apply a simple replication and our new replications. The 160 bit ID space and 2000 nodes are used to organize a p2p system. The size of the consistent set is 16 and the number of replicas is variable from 8 to 14. The target data durability for the availability based replication is decided by the number of replicas with mean node availability (0.5). That is, the target value is variable from 0.996 to 0.99994. The Poisson distribution is used to make the dynamic characteristic of nodes and the exponential distribution is used to assign on/off duration of a node. According to this Poisson distribution, the lifetime of 80% of total nodes is below 60% of total simulation time, that is, only 20% of total nodes has the reliable server-like profile. Recent research [6] measures the life distribution of the p2p nodes and shows the similar distribution, and we can tell that this distribution is similar to the real world.

Figure 5 shows the comparison of average data traffic per a node with various replication methods. As we expect, the simple replication needs much more data traffic to achieve the same data durability than our replications. The quorum based replication reduces the data traffic by about 40% and the availability based replication reduces the data traffic by about 60%. To find out the detailed effect of our replications, we separate the join data traffic from the leave data traffic. When a node joins, the affected nodes update their replicas and we call the needed traffic the join data traffic and when a node leaves, the needed data traffic is called the leave data traffic. In the Fig. 5 (b) and Fig. 5 (c), the simple replication uses the similar amount of traffic for join and leave. The main reason is that the replication is tightly coupled to the consistent set and in both type of the changes, the similar amount of compulsory copies is needed to support the simple replication. However, in our two replications, the join data traffic is less than leave data traffic. The replication set is loosely coupled to the consistent set and when a join occurs, a node can decide which it makes a new replica. According to this behavior, the replicas are interleaved on the replication set and we can reduce the number of compulsory copies when a new node joins.



(a) Total data traffic (b) Join data traffic (c) Leave data traffic

Fig. 5 Data traffic with various number of replicas.

However, like Fig. 5 (c), the quorum based replication needs similar amount of leave data traffic to the simple replication. This replication does not consider the node characteristics and some new replicas leave the system early after it is chosen by the other nodes. Figure 5 (c) shows that the availability based replication solves this problem and needs less leave data traffic than the quorum based replication. On the other hand, the availability based replication computes the data durability whenever the membership changes. According to this, it takes more care of selecting new replicas and it needs more join data traffic than the quorum based replication until the replication set is similar to the consistent set.

4. Related Work

SRDI (Shared Resource Distributed Index) on paper [10] is a smart method of data location to reduce data traffic under churn. It uses loosely coupled DHT which locate data on a RPV (Rendezvous Peer View). When a RPV shifts, a RPV only update its view, but the location of data does not change. If a user wants to find this data, a query walks through a RPV until it meets the data. However, to improve ability to retrieve the index even with a shifted RPV, data should be replicated on a RPV. Our paper focuses on this efficient data replication method.

The commercial p2p file sharing systems leave the data replication up to the popularity of the data. The popular data is replicated on many clients and the data durability of this data is very high. However, the unpopular data are stored on few clients and it is very hard to find this data because of very low data durability. To make the p2p storage system durable, the smart data replication methods is needed and the each inserted data is durable for any time.

In the paper [8], they store the replicas on the random nodes of the ID space and periodically check their availability. This behavior reduces the compulsory copies because the replication has no relation to the consistent set. However, this approach takes too much control traffic to keep the node availabilities of all replicas for every data on the system. Moreover, it does not use the consistent set and the change of the data durability caused by the node failure is detected slowly. The paper [9] shows that the erasure coding approach reduces the traffic of the replication by using the computing power. This approach is orthogonal to our approaches and we can use this coding with our replication methods.

5. Conclusions

We explore the new efficient replication methods to make the DHT based p2p storage system more durable. Our replication methods are loosely coupled to the consistent set and interleave the replicas on it. Because the consistent set updates the current state of nodes automatically, the data durability is updated immediately and automatically under churn. Moreover, the node availability is used to select more reliable replicas and more data traffic can be reduced when a node leaves. According to these behaviors, the DHT based p2p storage system with our replication methods achieves the high data durability with small data traffic. This can encourage that the DHT based p2p algorithms are applied to the durable storage system.

References

[1] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," Proc. ACM SIGCOMM 2001, pp.149–160, Aug. 2001.
[2] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," Proc. Middleware, pp.329–350, Nov. 2001.
[3] B.Y. Zhao, J. Kubiatowicz, and A. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," UCB Technical Report UCB/CSD-01-114, 2001.
[4] P. Druschel and A. Rowstron, "PAST: A large-scale, persistent peer-to-peer storage utility," Proc. HotOS VIII, May 2001.
[5] F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," Proc. SOSP 2001, pp.202–215, Oct. 2001.
[6] S. Saroiu, P.K. Gummadi, and S.D. Gribble, "A measurement study of peer-to-peer file sharing systems," Proc. MMCN 2002, 2002.
[7] K. Kim and D. Park, "Efficient and scalable client clustering for Web proxy cache," IEICE Trans. Inf. & Syst., vol.E86-D, no.9, pp.1577–1585, Sept. 2003.
[8] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G.M. Voelker, "Total recall: System support for automated availability management," Proc. NSDI 2004, pp.337–350, 2004.
[9] R. Bhagwan, S. Savage, and G.M. Voelker, "Replication strategies for highly available peer-to-peer storage systems," Proc. FuDiCo, pp.40–49, June 2002.
[10] H. Sunaga, T. Hoshiai, S. Kamei, and S. Kimura, "Technical trends in P2P-based communications," IEICE Trans. Commun., vol.E87-B, no.10, pp.2831–2846, Oct. 2004.