Predicting durability in DHTs using Markov chains

Fabio Picconi
University of Bologna, Italy
picconi@cs.unibo.it

Bruno Baynat
LIP6, Paris, France
bruno.baynat@lip6.fr

Pierre Sens
LIP6/Inria, Paris, France
pierre.sens@inria.fr

Abstract

We consider the problem of data durability in low-bandwidth large-scale distributed storage systems. Given the limited bandwidth between replicas, these systems suffer from long repair times after a hard disk crash, making them vulnerable to data loss when several replicas fail within a short period of time. Recent work has suggested that the probability of data loss can be predicted by modeling the number of live replicas using a Markov chain. This, in turn, can then be used to determine the number of replicas necessary to keep the loss probability under a given desired value.

Previous authors have suggested that the model parameters can be estimated using an expression that is constant or linear on the number of replicas. Our simulations, however, show that neither is correct, as these parameter values grow sublinearly with the number of replicas. Moreover, we show that using a linear expression will result in the probability of data loss being underestimated, while the constant expression will produce a significant overestimation. Finally, we provide an empirical expression that yields a good approximation of the sublinear parameter values. Our work can be viewed as a first step towards finding more accurate models to predict the durability of this type of systems.

1 Introduction

Large-scale distributed storage systems such as DHTs [1, 3, 4, 5, 2] can provide a low-cost alternative to expensive persistent storage solutions such as Storage Area Networks (SANs) and dedicated servers. By aggregating the hard disk space of a large number of computers, users can benefit from a large storage capacity at only a fraction of the cost of a centralized solution. Moreover, the geographical distribution of nodes avoids single points of failures, and makes the system more resilient to network partitions.

Persistence is achieved by replicating the same data object on several nodes, and ensuring that a minimum number of copies are available on the system at any time. Thus, whenever the contents of a node are lost, for instance due to a hard disk crash or a destructive operating system reinstall, the system regenerates the objects stored before the failure by transferring them from the remaining replicas. However, unlike local-area storage systems, wide-area systems suffer from low-bandwidth between replicas. The copies of an object are usually stored on geographically dispersed nodes to avoid correlated failures, so transfers between different countries or continents are not uncommon.

A low-bandwidth environment limits the system's ability to regenerate failed replicas, resulting in long repair times. For instance, if each node stores 100 GB, has a 1 Mbit/s connection to the Internet, and allocates one third of its total bandwidth to restore failed replicas, then repairing a crashed hard disk will take approximately one month. Long repair times, in turn, increase the probability of permanent data loss. Assuming random failures, there's a non-zero probability that all replicas of a given object will fail within a short period of time. Since it takes so long to restore the whole contents of a failed disk, the last replica of an object may be lost before the system can restore at least one copy of it. Adding more replicas decreases the chances of this hapenning, but does not solve the problem.

In practice, the number of replicas per object is chosen so that the probability of data survival is kept above a desired value. However, care must be taken when choosing the replication factor. An excessively high value will reduce the usable storage capacity and increase the overhead of new object insertions. Conversely, if the replication factor is too low, the permanent data loss rate will be high resulting in poor durability.

It has been recently suggested that object durability can be predicted by modeling the state of the system using a continuous-time Markov chain [6, 7]. Each state in the Markov chain represents the number of existing replicas for a particular object, and a state transition correspond to the loss or the regeneration of one replica. A transient analysis of the Markov chain yields the probability of reaching state i after some time t, and particularly state 0, which corresponds to the simultaneous failure of all replicas, i.e., permanent data loss.

Although the Markovian model is relatively simple, lit-

tle attention has been given to the choice of the model's parameters, i.e., the chain's transition rates. These rates must be estimated precisely in order to produce accurate predictions of the probability of data loss. Coarse estimations may result in the probability of data loss being underestimated. This, in turn, will lead the system's designer into choosing a replication factor based on poor model predictions.

Estimating the chain's transition rates accurately is not easy. For a system with k replicas, the model requires k failure rates and $k-1$ repair rates (one for each state i with $0 < i < k$). The major difficulty resides in estimating the repair rates, which depend on a myriad of factors such as the number of available replicas, the amount of data per node, the available bandwidth, and even the node failure rate.

This paper makes the following contributions. First, it shows that previous estimations of the model parameters based on a linear approximation will underestimate the probability of data loss. Second, it provides an empirical expression that approximates the sublinear growth of these parameters. Our results are obtained using a discrete-event simulator fed with synthetic failure traces.

The rest of this paper is organized as follows. Section 2 lists our assumptions, describes the Markov chain model, and discusses previous approximations of the chain repair rates. Section 3 evaluates the model's predictions through long-term simulations, and presents the empirical expression for the repair rates. Section 4 presents related work, and Section 5 concludes the paper.

2 Model

In this section we list our system assumptions, present some definitions used in later sections, and describe the Markov chain model used to predict data durability.

2.1 Assumptions and definitions

We consider a Distributed Hash Table composed of thousands of nodes connected to the Internet by low-bandwidth links, e.g., 1 Mbit/s links. Each object is associated with a unique key, and is replicated on k adjacent nodes which are close to the key in the DHT address space. This is a common replication scheme used by DHTs such as PAST [1] and OpenDHT [15]. We will assume an address space following a ring geometry as this is the easiest to visualize, although our results apply to other geometries such as XOR [3] and d-torus [5]. We assume that each node stores thousands of objects, for a total storage capacity of several tenths to hundreds of gigabytes per node.

The contents of a node may be lost due to a hard disk crash or a destructive operating system reinstall. We make no difference between the two, and we will use the terms node *failure* or *crash* to refer to same event, i.e., the loss of a hard disk's contents. We also assume that failures are random and uncorrelated.

We assume that failed hard disks are replaced by an empty disk quickly after the crash (i.e., within a few hours). We ignore this replacement time as it is negligible compared to the time needed to regenerate the disk contents. The node then starts regenerating the objects it stored before the crash using the following repair procedure: first, since replicas are stored on ring neighbors, the node determines which objects are to be restored by querying its neighbors. Then, the node starts transferring the objects sequentially from the remaining replicas. This is basically what existing DHTs do to regenerate replicas after a crash.

We assume highly stable nodes, i.e., that the disconnection and churn rate are low. Contrary to P2P file-sharing systems, which are characterized by high churn [8, 9], storage systems must rely on nodes which stay connected for long periods of time (i.e., weeks or months) in order to guarantee data persistence [10]. These could be, for example, the workstations of a corporate network (in a system that federates several networks), users who leave their P2P client running continously, or residential set-top boxes equipped with hard disks and running a P2P storage service. Therefore, we assume that temporary node disconnections only occur during a network partition or a maintenance operation. Because of these reasons, churn and disconnections will not be considered in our analysis. Also, since we assume high availability, our analysis will only consider systems that use replication, rather than erasure codes, as the former has been shown to be more efficient for highly available nodes [17].

Finally, we list some important definitions that will be used in the following sections:

- MTBF. Mean time between failures of a given node.
- b. Average number of bytes stored per node.
- bw_{max}. Maximum bandwidth per node allocated to regenerate lost replicas.
- θ. The value of θ gives the ratio between the MTBF and the minimum time needed to restore the contents of a hard disk. It is defined as follows:

$$\theta = \frac{MTBF}{b/bw_{max}}$$

2.2 Markov chains

In order to estimate data durability, we model the state of a data object using a continuous-time Markov [6, 7]. Each state in the chain represents the number of replicas for a particular object that exist in the system at time t after its insertion into the DHT. Therefore, the chain has $k+1$ states, 0 to k, where k is the replication factor.

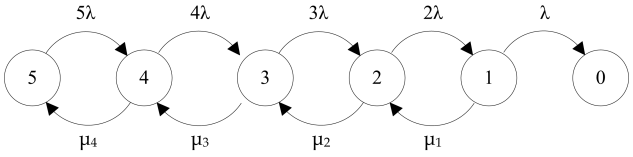

Figure 1. Markov-chain model.

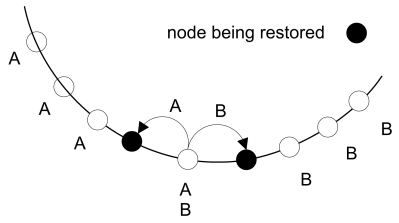

Figure 2. Objects A and B are restored from the same source node.

Figure 1 shows the Markov chain for $k = 5$. State transitions correspond to replicas being lost or restored. Once state 0 is reached, there are no more copies left and the object is lost. The probability of being in state i as a function of the time t can be obtained by solving a system of $k+1$ linear differential equations. The probability of losing the object permanently is simply the probability of reaching state 0 at time t.

Failure rates, i.e., transitions to lower states, are caused by hard disk crashes or system reinstalls, so they can be approximated using the disk MTBF provided by the manufacturer or the destructive reinstall rate observed on the real system. As usual, we will assume that the time between failures of the same node is exponentially distributed with mean MTBF$= \lambda^{-1}$ [11, 6, 7]. Thus, if each disk fails with MTBF, then the time between two failures in a set of i independent disks becomes MTBF/$i = (i\lambda)^{-1}$. This yields a failure rate $i\lambda$ from state i to state $i-1$.

The repair rate μ_i, i.e., from state i to state $i+1$, is much harder to determine. Intuitively, it corresponds to the inverse of the mean time needed by the system to restore a *single* copy of the object when $k-i$ replicas are missing. However, the average time needed to regenerate *one* replica is not independent of the number of replicas being repaired. As the number of failed replicas increases, fewer sources become available to download from. This means that two or more nodes may contact the same source, congesting its upstream link and decreasing the average transfer rate. This is the main reason why the repair rates μ_i do not grow linearly with the number of missing replicas.

Another interesting point is that this type of congestion at the source may also occur when only a single replica of the object is missing. Figure 2 shows a case in which only one replica is missing for both objects A and B, but both nodes being restored contact the same source concurrently (since it is chosen randomly), resulting in a halved transfer rate. The probability of this happening depends strongly on the value of θ. If θ is high, then failures will be rare and the probability of two or more nodes downloading from the same source will be very low. However, for low values of θ (i.e., when the value of the MTBF is comparable to the hard disk restore time), then the probability of such a congestion will not be negligible.

2.3 Previous estimations of μ_i

Chun et al. [7] have suggested an approximation of the repair rates μ_i based on a constant expression. Their estimation is as follows: if each node stores b bytes of data in its hard disk, and has allocated a maximum repair bandwidth of bw_{max} bytes/sec, then the time needed to restore a crashed hard disk is $T_r = b/bw_{max}$. The repair rates are then estimated using $\mu_i = 1/T_r = bw_{max}/b$.

In a different approach, Ramabhadran et al. [6] suggest that the repair rate μ_i grows linearly with the number of missing replicas: $\mu_i = (k-i)\mu$. However, the authors do not provide any expression for μ, as they consider it to be a tunable system parameter, dependent on how aggressively the system reacts to failures.

As we will see in the next section, none of these expressions yields satisfactory results. The reason is that the actual repair rates are neither constant nor linear, but grow sublinearly with the number of replicas.

3 Approximating the real μ_i

In this section we will first use the simulator to measure the actual values of μ_i, and then propose an empirical expression to approximates these values.

We use a discrete-event simulator that implements a simplified version of the PAST protocol [1]. Each node has a unique identifier in a ring address space. Data objects are replicated on the k nodes which are closest to the object key. We assume a symmetric repair bandwidth of 1.5 Mbit/s, as was previously done by other authors [7]. Each node stores b bytes, which can vary from tenths to hundreds of gigabytes of data, according to the experiment. In all cases the data stored by each node is divided into 1000 objects, so each object has a size $b/1000$ bytes. We assume high node availability (cf. Section 2.1), so temporary node disconnections are ignored by our simulator.

All our simulations use a DHT of 100 nodes, as our simulations show that increasing the network size does not qualitatively change the results. The reason is that objects are replicated on adjacent nodes on the ring, so restoring a

node's hard disk only affects a small number of other nodes. We generate synthetic traces of failures by obtaining node inter-failure times from an exponential distribution of mean MTBF [7]. Whenever the last replica of an object is lost, the simulator records the time elapsed since the insertion of that object into the DHT. In addition, each time an object is permantently lost we reinsert another object of the same size, thus ensuring that the amount data in the DHT remains constant during the experiment.

Unless otherwise noted, we use an MTBF of two months. Although this is small for a hardware failure rate, it is not far from the failure rate observed in a study conducted on Planetlab nodes [12]. Also, the durability of a system depends on the key parameter θ, i.e., the ratio between the MTBF and the node capacity. For instance, a system with $\theta = 10$ will exhibit the same durability whether it stores 100 GB per node with a MTBF of 2 months, 500 GB per node with an MTBF of 10 months, or any other equivalent combination of b and MTBF. By choosing a MTBF of 2 months and varying the node capacity b between 50 and 500 GB per node, we can test our model for $2 \leq \theta \leq 20$, thus covering the configurations most likely found in real systems.

3.1 Measuring μ_i

The real values of μ_i are measured as follows: for each object, the simulator measures $T_{a,i}$, which corresponds to the total time during which exactly i replicas are available on the network. We also determine $n_{r,i}$, the number of times a replica of the object is repaired, given that i replicas were available just before the repair is complete. We then average those values for all objects in the system, obtaining $\overline{T}_{a,i}$ and $\overline{n}_{r,i}$. Finally, the mean repair rate from state i is measured as $\mu_i = \overline{T}_{a,i}/\overline{n}_{r,i}$.

For convenience of presentation, we introduce a new notation and express μ_i as a function of the number of missing replicas $m = k - i$. In other words, μ_i and μ_{k-m} refer to the same transition rate. Figure 3 shows the measured values of μ_i as a function of the number of missing replicas m, for $\theta = 4$ and different values of k. For each k, the values are normalized to μ_{k-1}, i.e., $m = 1$. The curve shows that μ_i starts with a linear growth, but then increases sublinearly for higher values of m. As explained in Section 2.2, this is due to the fact that the upstream bandwidth of the available replicas must be shared across an increasing number of downloaders as more replicas are lost.

3.2 Approximating function

Although an expression for μ_i is hard to obtain theoretically, the values measured by simulation seem to follow a pattern. The curves shown in Figure 3 start with a linear growth and slope μ_{k-1}, and then seem to reach a plateau for high values of m.

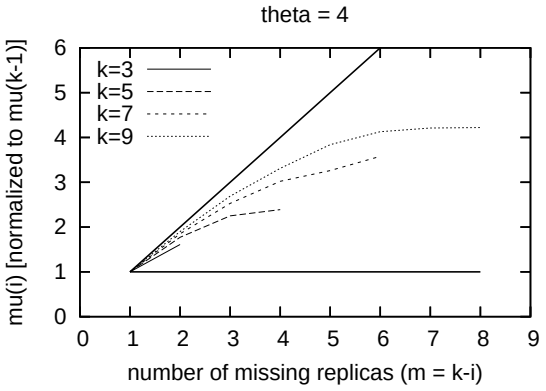

Figure 3. Repair rates μ_i measured by simulation, as a function of the number of missing replicas $m = k - i$, and normalized to μ_{k-1}.

This pattern suggests that one could derive a general expression for μ_i from the measured values. This expression could be used to approximate μ_i for others combinations of k and θ without resorting to further simulations. In other words, we propose an empirically-derived function $f(m)$ that approximates the values of μ_i:

$$\mu_i = f(k - i) = f(m)$$

Given the measured values of μ_i, we propose to use a function $f(m)$ with the following characteristics:

$$f(1) = \mu \quad (1)$$

$$f'(1) = \mu \quad (2)$$

$$\lim_{m \to \infty} f'(m) = 0 \quad (3)$$

$$f(k-1) = \frac{k+1}{2}\mu \quad (4)$$

The factor $(k + 1)/2$ of equation 4 has been chosen empirically so that $f(m)$ provides a close approximation of the value of μ_i measured for the different combinations of k and θ used in the previous sections.

Note that the value of μ corresponds to $m = 1$, that is, to the repair rate when only a single replica is missing. Thus, we have $\mu = \mu_{k-1}$. The value of μ of a real system can be easily approximated by measuring the averate time $\overline{t_r}$ needed to restore an object after a crash, and then calculating $\mu = 1/\overline{t_r}$. We have also derived an analytical expression that approximates μ given the system parameters MTBF, b, and bw_{max}. We omit the description of this expression for space reasons.

Among the large number of analytical functions that match the previous characteristics, one possibility is to use an exponential to construct $f(m)$. This yields an expression

m	$k=3$	$k=5$	$k=7$	$k=9$
1	15% 10% 5%	13% 12% 6%	14% 13% -	16% 13% -
2	1% 10% 17%	8% 7% 3%	12% 11% -	15% 12% -
3	N/A	2% 1% 1%	11% 10% -	15% 13% -
4	N/A	3% 10% 4%	7% 8% -	13% 12% -
5	N/A	N/A	1% 3% -	10% 12% -
6	N/A	N/A	9% 3% -	6% 9% -
7	N/A	N/A	N/A	1% 3% -
8	N/A	N/A	N/A	1% -2% -

Table 1. Approximation error $e(m) = |\mu_{k-m} - f(m)|/\mu_{k-m}$, where μ_i is the value measured by the simulator. Each triplet corresponds to $\theta = \{2, 4, 10\}$. A missing value means that the state was never reached during the whole simulation.

of the following form:

$$f(m) = \alpha(1 - e^{-(m-1)\mu/\alpha}) + \mu \quad (5)$$

This function satisfies equations 1 to 4. The constant α can be obtained by applying equation 4 and solving numerically, as α is both inside and outside the exponent. It turns out that this function provides a good approximation of the transition rates measured by simulation. Table 1 shows the error between the value produced by $f(m)$ and the measured one for various combinations of θ and k. For all measured values of μ_i the difference is always below 17%, with an average error of less than 10%.

The fact that the empirical function $f(m)$ fits the measured curve so well suggests that an analytical expression for μ_i may not be so hard to derive. We are, in fact, currently working on obtaining the values of μ_i analytically.

3.3 Predicted loss probability

Finally, we compare the probability of object loss predicted by the Markov chain when using the various approximations of μ_i described in the previous sections. Figure 4 shows the predicted and simulated loss probabilities for two replication factors, $k = 5$ and $k = 7$, and $\theta = 4$ (i.e, 250 GB per node). The curves show the predictions obtained using the constant and linear expressions for μ_i suggested by other authors (see Section 2.3), as well as that obtained using the approximation $\mu_i = f(k - i)$ described in the previous section.

The curves show that a linear expression for μ_i produces a prediction which underestimates the real data loss rate. This is not surprising, as the linear form overestimates the repair rate of the system (cf. Figure 3). Conversely, assuming that μ_i is constant greatly underestimates the system's repair rate, resulting in a predicted loss rate which is orders of magnitude above the real one.

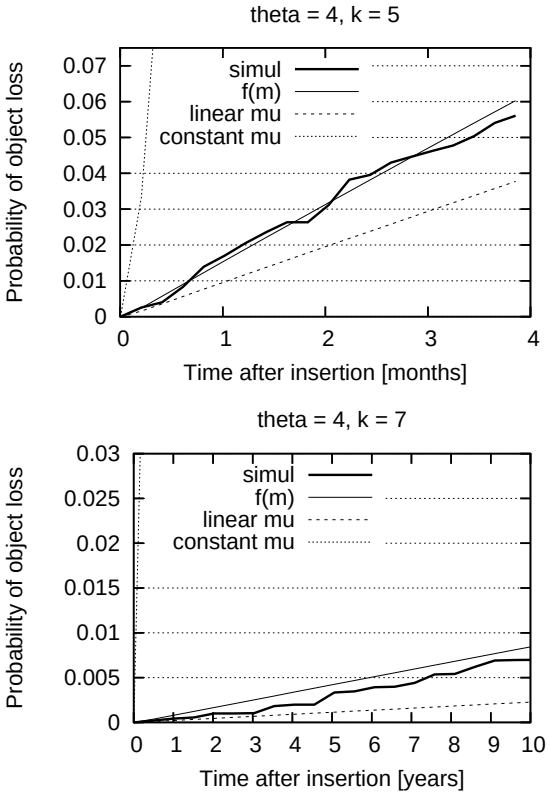

Figure 4. Probability of object loss measured by simulation, and predicted using $\mu_i = f(k - i)$ and our linear approximation of μ_i, for $k = \{5, 7\}$ and $\theta = 4$.

The prediction obtained using $\mu_i = f(m)$ is very close to the measured loss rate, suggesting that the approximating function produces good results. It is important to note that we are not simply feeding the measured rates μ_i into the Markov chain (in which case the measured and predicted loss rates would always match), but we are using the rates approximated by $f(m)$. In other words, we are checking that our function $f(m)$, besides from approximating μ_i well, also produces good loss rate predictions.

Although we do not show it for space reasons, $f(m)$ also produces good predictions for other combinations of k and θ.

4 Related work

Early work on tolerance to hard disk failures led to the design of RAID systems [11]. Peer-to-peer storage differs significantly from RAID in that replicas are geographically dispersed and connected through low-bandwidth links.

Durability in P2P storage systems was first studied by

Blake and Rodrigues [10], who presented a lower bound on node lifetime as a function of node capacity and bandwidth. They concluded that durability in wide-area systems requires long node lifetimes because of the long repair times.

Ramabhadran et al. [6] and Chun et al. [7] independently suggested to use Markov chains to predict the probability of data survival in DHTs. Unfortunately, both studies lack an accurate estimation of the chain transition rates. The former assumes that the repair rate is a tunable parameter, but thus does not provide a maximum value given the mean capacity, bandwidth, and MTBF. The latter suggests a simple expression for the repair rates, but this paper has shown that their approximation is too coarse and produces inaccurate results.

Finally, Bhagwan et al. [14] have proposed a system that dynamically varies the replication factor to achieve high availability. However, their work focuses on availability and transient node disconnections, not on permanent replica failures.

5 Conclusions

In this paper we have focused on finding an accurate prediction of data durability in DHTs. Recent work has suggested that this can be achieved by modeling the system's behavior using Markov chains. However, our experiments show that the model parameters suggested previously produce inaccurate predictions of the probability of data loss. In order to obtain a more accurate estimation of these parameters we have analyzed the behavior of a simple DHT protocol and showed that the repair rates grow sublinearly with the number of replicas. We have also proposed an approximating function for those rates that can be used for several combinations of system parameters (MTBF, node bandwidth and storage capacity). We have validated our approximating function through simulation, and showed that it yields far more accurate predictions than those suggested by other authors. Future work may include producing a finer model of the system, for instance, on that takes into account temporary node disconnections.

References

[1] A. Rowstron and P. Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *Proc. of SOSP*, 2001.

[2] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu. OpenDHT: A Public DHT Service and Its Uses. In *Proc. of SIGCOMM*, 2005.

[3] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-peer Information Systems Based on the XOR Metric. In *Proceedings of IPTPS*, 2002.

[4] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *Proc. of SOSP*, 2001.

[5] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. of SIGCOMM*, 2001.

[6] S. Ramabhadran and J. Pasquale. Analysis of long-running replicated systems. In *Proc. of INFOCOM*, 2006.

[7] B. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, F. Kaashoek, J. Kubiatowicz, and R. Morris. Efficient Replica Maintenance for Distributed Storage Systems. In *Proc. of NSDI*, 2006.

[8] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. of MMCN*, 2002.

[9] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proc. of SOSP*, 2003.

[10] C. Blake and R. Rodrigues. High availability, scalable storage, dynamic peer networks: Pick two. In *Proc. of HotOS*, 2003.

[11] D. Patterson, G. Gibson and R. Katz. A case for redundant arrays of inexpensive disks (RAID). In *Proc. of SIGMOD*, 1988.

[12] F. Dabek. A Distributed Hash Table. Ph.D. thesis. 2005.

[13] P. Tam. A Physicist's Guide to Mathematica. Elsevier, 1997.

[14] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. Voelker. Total Recall: System support for automated availability management. In *Proc. of NSDI*, 2004.

[15] S. Rhea. OpenDHT: A Public DHT Service. Ph.D. thesis. 2005.

[16] B. Zhao, L. Huang, J. Stribling, S. Rhea, A. Joseph, and J. Kubiatowicz. Tapestry: A Resilient Global-Scale Overlay for Service Deployment. In *IEEE Journal on Selected Areas in Communications*, Vol 22, No. 1, January 2004

[17] R. Rodrigues and B. Liskov. High Availability in DHTs: Erasure Coding vs. Replication. In *Proc. of IPTPS*, 2005.