Model and Evaluation of Redundant Data Organization for High Availability in Structured Overlay Networks

Xu Guangping[1,2], Ma Yong[1], Ma Wenhui[1],Wang Gang[1], Liu Xiaoguang[1], Liu Jing[1*]

1. Information Technology Science College, Nankai Universitity, Tianjin, China
2. Computer Science and Technology School, Tianjin University of Technology, Tianjin, China
Email: xugp2008@yahoo.com.cn

Abstract

The paper presents a hierarchical organization of two storage redundancy schemes to improve data availability caused by churn in structured overlay networks. To mask or hide the high churn from the portion of short-lived but frequent churn peers and permanent failure peers, we use replication among the nodes in a certain interval that can be considered as a virtual node. Then a set of virtual nodes that cooperatively provide guaranteed over the networks with erasure coding. We present the analysis of the behaviors of single one virtual node and cluster of virtual nodes. According to the stochastic models of the behaviors, the data availability under churn is presented. We also give some quantitative analysis based on one empirical trace dataset.

1. Introduction

Nowadays networks are growing towards an ever increasing scale and heterogeneity and becoming extremely complex. The arrival and departure of nodes occur concurrently and frequently in networks. To manage data in the complex networks, various applications usual employ structured overlay network technologies to organize and manage nodes. Structured overlay network is a high logical level network architecture built on the existing networks. The benefits of the structured approach are resilience to node failures. Routing and location of data objects can be succeed in the face of dynamic network changes and node failures, as a result of the design for resilience. And the networks offer the promise of systems that automatically scale in capacity as the number of users increases and yet are extremely robust, automatically adapting to failures of peers as well as to changes in usage patterns.

In these loosely organized networks, the tremendous growth in storage and communication bandwidth of autonomous computers can supply various services in which make the resources available to other peers. Storage service is an important part for the new computing paradigm. For example, file system, information dispersal system, distributed media system and so on are the typical applications. But due to the dynamic and scalable nature of the networks, it is a foundational and challengeable issue to ensure data availability in deploying various applications. Usually, high availability in overlay networks usually utilizes some forms of data redundancy including replication and erasure coding. Normally, replication is a form redundancy by creating multiple copies. With replication approach, an object (data item) is duplicated into multiple replicas and distributed to different peers. An object can be available as long as at least one replica is available. Erasure coding [11-13] as the other typical form of redundant data storage yields much higher probabilities of recovery and offers lower storage costs compared to replication schemes. In the case, an object is divided into n blocks and distributed to different peers. As long as at least m ($m<n$) blocks are available, the original object can be recovered.

Some probability models and analysis were discussed in [6, 7, 15] involving the parameters of redundant data, such as the average availability of peers, storage overhead and the required data availability. According to these models and analysis [15], erasure coding is not always preferable to replication at the situation that availability of peers is relative low especially. The decision is not made easily

* The work in part is sponsored by National Science Foundation of China (No.90612001), Education Ministry Doctoral Research Foundation of China (No.20070055054), Science and Technology Development Plan of Tianjin (No. 043185111-14), Education Ministry Doctoral Research Foundation of China(No.20070055054), 863 program (No. 2008AA01Z401) and Tianjin Natural Science Foundation (No. 08JCYBJC13000).

in the complex and heterogeneous networks. The work gives our motivation to combine replication with erasure coding.

In this paper, we first introduce a hierarchical organization of redundant data in section 2, as our work in [25]. With sufficient redundancy with many peers, at any moment couple of peers will be in the system to make a given data object available with high probability cooperatively. When a peer is available, the data stored on it contributes to the overall degree of redundancy and increases data availability. Many peers contribute to store the related data and takeover some node failures for the availability of data in systems. We focus on model and evaluation the hierarchical redundant data to achieve high data availability in section 3 and 4. Finally, we briefly discuss the future work.

2. System

In structured overlay network, the whole identifier space is split into equal-size intervals. A virtual node could be taken as the abstraction of multiple physical nodes in the equal-size interval in which the physical nodes can cooperatively accomplish routing and storage requests. Each interval (i.e., virtual node) is maintained by multiple nodes at the same time and each peer in every virtual node contributes storage space and cooperatively makes stored data persistent and available.

We categorize a variety of placement strategies into two types: Locally Neighboring Placement (LNP) and Globally Rehashing Placement (GRP). The classification is based on the distribution fashion of redundant data (replicas with replication and blocks with erasure coding of an object) among peers in systems. In LNP redundancy objects are placed on the neighboring nodes of the owner node, which is a node responsible for the objects according to consistent hashing. The neighbors can be successor nodes, predecessor nodes or entry nodes of routing table. For example, [1, 18] employ a successor-list scheme in which an object is replicated on the successive nodes with closest to the owner. In contrast, in GRP the responsible node of each redundant object is determined by a rehash function in the global space. The key of each replica can be obtained by re-hash function and then the corresponding node can be determined. For example, [19, 24] provide a symmetric replication scheme in which the identifier space is partition into equivalence classes and an object is duplicated on each of the nodes in an equivalence class. In the paper, we employ the two placement strategies to distribute redundancy objects (replicas within a virtual node by LNP and blocks of by GNP) in the identifier space.

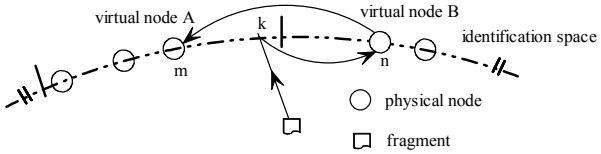


Figure 1. A block is mapped to an identifier *k* belonging to virtual node *A*. But the successor of the identifier is physical node n belonging to virtual node *B*. In the case, the request message will be redirected to its predecessor.

We utilize replication and erasure coding to place redundancy objects in the structured overlay network space. On the whole, with erasure coding each data object is divided into *m* blocks and then recoded into *n* (*n*>*m)* blocks and stored into different virtual nodes at virtual node layer. We place *n* encoded blocks into different virtual nodes according to GRP. Within a virtual node, a block is replicated into its physical nodes. The replicas of each block in a virtual node are distributed by LRP.

Data operations include the following interfaces: block operation interfaces (PUT_BLK, GET_BLK) and object operation interfaces (PUT_OBJ, GET_OBJ and REP_OBJ). If a peer decides to put an object (PUT_OBJ) that consists of *n* independent blocks, then the peer issues the PUT_BLK message for each block with its identifier *k*. Once at least *m* messages are acknowledged, the block is putted into the system. Otherwise, the peer tries a certain threshold times. Similarly, if a peer tries to get an object, then it issues the GET_BLK message for each block. Once *m* responses are received, then the object can be recovered. Otherwise, the peer tries threshold times. If the destination node is routed, then these operations will be performed and response will be acknowledged to the requestor. PUT_BLK tries to store a block in a virtual node; GET_BLK tries to retrieve a block if it exists in a virtual node.

When a node joins an active virtual node, it announces itself to all other nodes responsible for the same interval. System should recognize two cases: fresh join and return. Data synchronization should be taken among the nodes in the interval in both cases. All data associated in the virtual node is replicated to the node if it is a fresh one; and if it is back after a temporary failure period, it should exchange data that each other does not have, that is, synchronize offline replicas after going online again. When any node

leaves the system for the process is inactive, the standard stabilization routine will be performed. The predecessors and successor will be informed, and afterwards, inconsistent routing table entries are identified and updated using the periodic maintenance routine [1].

Data routing and location requests can be processed almost the same as the specification of DHT among the physical nodes in the identification space but few changes. Figure 1 presents that when a block is mapped to an identifier *k* belonging to virtual node *A* but the successor of the identifier is physical node *n* belonging to virtual node *B*. Thus, the routing and location for the block to store and access may be redirected to the predecessor node *m* by node *n*. Even thought in the procedure the redirected hop may be an incidental expense, it may save one more hops when routing and location for a block reach the corresponding virtual node rather than the owner node. The related procedures are executed whenever data operation messages arrive at the related nodes. As a variation of Chord, the routing table can be the same as finger table in the protocol. Indeed the protocol can combine into the other DHTs as well.

The routing and resolution algorithm for data requested message (Figure 2) is crucial in the system design. Upon receive a message, a node first checks to see whether the requested identifier *k* falls within the range of the virtual node. If so, the node determines and executes the action by resolving the message. Otherwise (i.e., the virtual node does not cover the identifier), then one of the following cases may happen. The message is redirected to the predecessor node m by node *n* in one case as shown in figure 2 (case 1). If the redirected node is not responsible for the block, the virtual node is unavailable and the message will be abandoned (case 2). In the other case, the routing table is used and the message is forwarded to a node that is the closest to the identifier (case 3).

4. Model

In the section, we present the analysis of the behaviors of single one virtual node and cluster of virtual nodes as result of churn in structured overlay networks.

It was observed that while a little amount of short-lived peers join and leave the system at such a high rate that they constitute a relatively large portion of sessions at any point of time [8-10]. Those short-lived and churn-frequent nodes will have much negative impact on data availability. The impact can be reduced by the convergence behavior of nodes in a virtual node. Thereby the replication model could not be constrained to much heterogeneous nodes. That is, virtual node weakens even masks the behaviors of those churn-frequent nodes, i.e., those temporary failed nodes. As for the impact from those permanent failed nodes and fresh nodes, it could also be reduced to some extents.

```
//find the destination to execute the operation in msg by n
//according to the identifier k of the block in msg
n.find_destination(msg){
  if (msg.k )∈ n.interval()
      //invoked the operation of message by n
      return (function *)msg.oper;
  if(msg.redirect ){// (case 2)
    //if the message is redirected, virtual node has failed.
       error("could not routing the msg");
       return(-1);
  }
  if(msg.k < n.interval().lowbound
       && msg.k ∈  (n.id, n.successor) ) {//case 1
    msg.redirect = TRUE;
    p = n.predecessor();
    forward(msg, p);//send msg to the predecessor of n.
  }
   else {//case 3
        //search the closest node in routing table as in [1]
      n'= closest_preceding_node(msg.k);
        forward(msg, n');
      }
}
```

Figure 2. The routing and resolution algorithm for data requested message.

The on-line/off-line alterations of an individual node can be modeled as an alternating renewal process, having an on process and an off process. We assume that physical nodes behave independently of each other and that each process $\{Z_i(t)\}$ is independent on each other. The collective effect of multiple nodes in one virtual node can be also modeled as the superposition of multiple alternating renewal processes. We are primarily interested in the active replica degree, which can be defined as the number of active nodes in one virtual node (the size of a virtual node is *s*-bit length) at any time *t*. Let $\{Z(t)\}$ denote the evolving process of the active replica degree, i.e., the number of active peers in system at time *t*, and the process is a regenerative process with the regeneration point defined at an arrival time.

Let T_k be the amount of time in a cycle with *k* peers present and $C = \sum_{i=0} T_i$. Let p_k denote the limiting

probability that there are k peers in one virtual node at any time. In the count process $\{Z(t)\}$ produced by the M/G/ ∞ queue model, which can be considered that peers arrive at an infinite server queue according to a Poisson process and the on-time of each peer can be general. From queuing theory, if the queue is stable, then $p_k = \lim_{t\to\infty} P\{Z(t) = k\} = E[T_k]/E[C]$ and the expected value $E[C]$ is finite. Thereby, the availability of blocks in one virtual node is given by $v=1-p_0$. The probability of p_k can be reduced to the number distribution of active peers in one virtual node.

Given a system in the stable, according to the parameters n and m of erasure code, an object will be encoded into n fragments and distributed in n different virtual nodes. There are $n+1$ states for each item, $\{0,...,n\}$. State k is the state that k virtual nodes are alive and $(n-k)$ virtual nodes are unavailable. Obviously if an object is in any state i ($i<m$), it can't be reconstructed. Let A denote the available state set and $\overline{A}$ the unavailable state set. Therefore, with the (m, n) erasure coded scheme, the sets $A = \{m,...,n\}$ and $\overline{A} = \{0,...,m-1\}$ are hold in the context. The process is said to be *up* if it is in an available state and *down* in an unavailable state.

As analyzing the convergence behavior of multiple nodes in single virtual node, we can suppose that the change of a virtual node is independent and identical process with each other and data availability process changes states in accordance with a Markov chain having transition probabilities $P_{ij}, i, j \in \{0,1,...,n\}$. The states of Markov chain are the number of active virtual nodes. We concern about the rate of the process from up to down (the rate of breakdowns) and the average up-time and down-time in one cycle period. According to limiting probabilities for Markov chain [21], there exists a limiting probability π_j that the process will be in state j after a large number of transitions, and the value is independent of the initial state. Now for $i \in A$ and $j \in \overline{A}$ the rate at which the process enters state j from state i is $\pi_i P_{ij}$. And so the rate at which the process enters state j from an available state i in A is $\sum_{i\in A} \pi_i P_{ij}$. Hence, the rate at which it enters any unavailable state from any available one (which is the rate at which breakdowns occur) is $\sum_{j\in\overline{A}} \sum_{i\in A} \pi_i P_{ij}$.

Now let u and d denote the average up-time and down-time separately in the process. Thereby, the transition frequency between any available state and unavailable one is $1/(u+d) = \sum_{j\in\overline{A}} \sum_{i\in A} \pi_i P_{ij}$. The average percentage of up-time in one cycle is $\sum_{i\in A} \pi_i$. Since the process is up on the average u out of every $(u+d)$ time units, the proportion of up-time is given by $u/(u+d) = \sum_{i\in A} \pi_i$. Therefore, the average up-time is

$$u = \sum_{i\in A} \pi_i / \sum_{j\in\overline{A}} \sum_{i\in A} \pi_i P_{ij} \quad (1)$$

and down-time is

$$d = \sum_{i\in\overline{A}} \pi_i / \sum_{j\in\overline{A}} \sum_{i\in A} \pi_i P_{ij} \quad (2).$$

5. Experiments

According to the trace dataset used in PlanetLab [20], it consists of pings sent every 15 minutes between all pairs of 200~400 PlanetLab nodes from January, 2004 to June, 2005 (i.e., $4.557*10^7$ s). In our simulation, the trace was removed each period of downtime when less than half the average number of nodes up as done in [8]. In the period, the system scale, i.e. the number of nodes, amounts to 669 nodes in the system. The session distribution (25130 sessions) has heavy-tailed property suggested in many studies, such as Pareto[14-15], Weibull distribution[9], which reveals the heterogeneous behaviors of nodes.

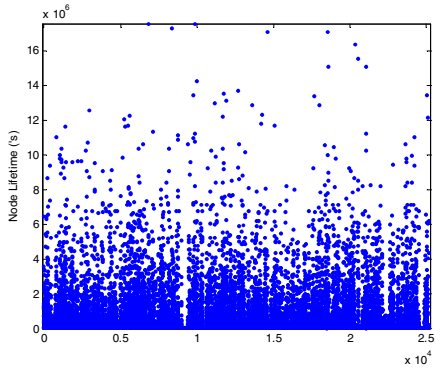
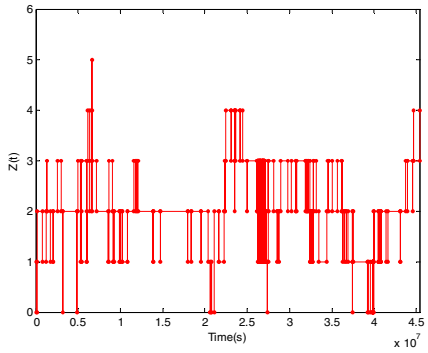


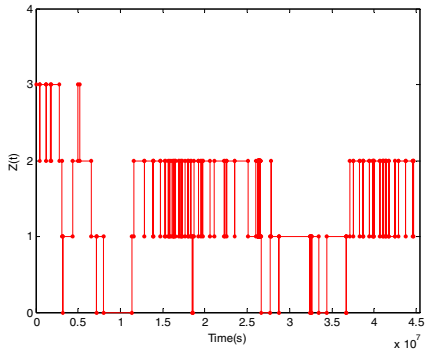
Figure 3. The session distribution in the trace.

As intended, rather than tangle with various behaviors of single peer, the availability of each block is analyzed by the converge behavior of virtual node to mask the churn-frequent nodes. However, the cycle times are too heterogeneous to estimate exactly the availability as shown in figure 3.

The sample paths of $\{Z(t)\}$ are shown for virtual node size $s=8$ *bits* in Figure 4(a) and $s=6$ bits in Figure 4(b). In the paths, a regeneration cycle contains an up period with at least one peer present and a down period with no one present. As an important observation, the coverage behavior of each virtual node can mask failures effectively including temporary and permanent failures for the down periods are relatively little time, about 10^2~10^3s.



(a) s=8



(b) s=6

Figure 4. The sample path of {*Z(t)*} in a virtual node for different size of virtual node.

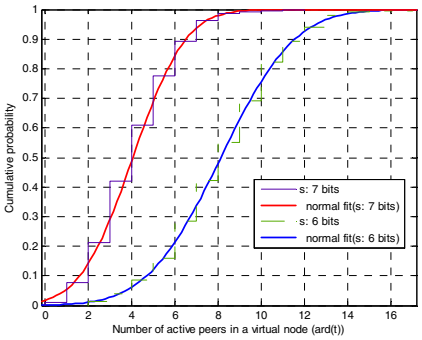
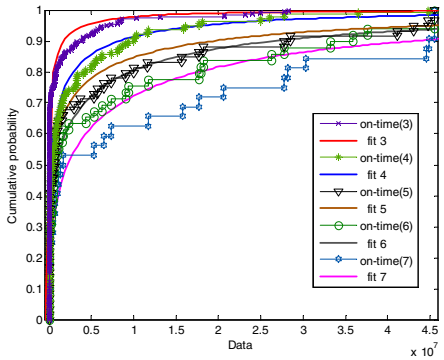
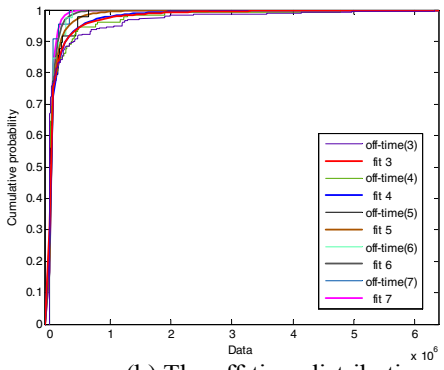


Figure 5. The number of active peers in a virtual node tends to a normal random variable.

The number of active peers in a virtual node tends to a normal random variable observed as shown in Figure 5 for different sizes of virtual node, *s=7* and *6* respectively. The distribution parameters are (4.06, 1.91) for *s=7* and (8.13, 2.67) for *s=6*.



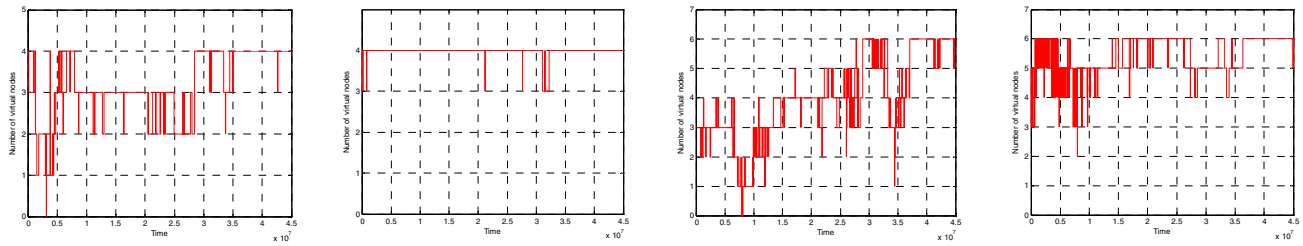
(a) The on-time distribution



(b) The off-time distribution

Figure 6. The on and off time distributions of a block replicated in a virtual node.

The on-time and off-time of a block replicated into one virtual node. In experiments, the size of virtual node is set different values, 3~7 bits. From the data as figure 6(a), the convergence of heterogeneous nodes is much more effective as the size of virtual node increases. The quantity of off-times is similar as observed in figure 6(b). Amount of short-lived peers join and leave the system at such a high rate that they constitute a relatively large portion of sessions at any point of time. Those short-lived and churn-frequent nodes will have much negative impact on data availability. However, the impact can be reduced by the convergence behavior of nodes in a virtual node. Thereby the replication model could not be constrained to much heterogeneous nodes. That is, virtual node weakens even masks the behaviors of those churn-frequent nodes, i.e., those temporary failed nodes. As for the impact from those permanent failed nodes and fresh nodes, it could also be reduced to some extents.



(a) s=3, n=4 (b) s=4, n=4 (c) s=3, n=6 (d)s=4, n=6

Figure 7. The availability results of virtual node cluster for different parameters

By exploring a large number of different settings with the parameters, we present the average availability of an object as shown in Figure 7. In the experiments, the parameters are set as the following: *s=3, 4* bits and *n=4, 6*. The choice of parameter *n* can be determined constrained by the requirements of the availability and related costs. According to the result of figure 7(a)~(d), the tradeoff values of *n* could be set. The system designers must choose the moderate values for *(s, m, n)* to make the data more available and reliable. Due to space limitation, we leave it as further work.

5. Discussion

The up-time of an object is dependent on the parameters of the redundancy layers when it is put into the system which includes the size of virtual node s and erasure coding parameters *n* and *m*. The tradeoff of parameters between the two redundancy layers is flexible and vital. To achieve the same extents of available, the choices of the parameters are crucial. In one extreme, the size of virtual node s is to the greatest extent and erasure code parameters can be relaxed. In the other extreme, n is a larger number and m is close to 1, thereby the virtual node size s is relaxed. However, the former will consume much to maintenance synchronization and the latter will be degraded to replication that consumes much storage space.

In future work, we plan to model and analyze maintenance mechanism including correlative churn impact on redundant data. The parameter needs further tradeoff between performance and cost.

6. References

[1] I. Stoica et al. Chord: A scalable peer-to-peer lookup service for internet applications. In Proc. of ACM SIGCOMM, 2001.

[2]A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proc. of IFIP/ACM International Conference on Distributed Systems Platforms, 2001.

[3]F. Dabek et al. Wide-area cooperative storage with CFS. In Proc. of SOSP, 2001.

[4]J. Kubiatowicz et al. Oceanstore: An architecture for global-scale persistent storage. In Proc. of ASPLOS, 2000.

[5] Ranjita Bhagwan et al. Total recall: System support for automated availability management. In NSDI, 2004.

[6]R.Bhagwan, S. Savage, and G. M. Voelker. Understanding availability. In Proc. of IPTPS, 2003.

[7]Rodrigo Rodrigues, Barbara Liskov. High availability in DHTs: Erasure coding vs. replication. In Proc. of IPTPS, 2005.

[8] P. Brighten Godfrey, Scott Shenker, and Ion Stoica. Minimizing Churn in Distributed Systems. In Proc. of SIGCOMM, 2006.

[9]Daniel Stutzbach, Reza Rejaie. Understanding Churn in Peer-to-Peer Networks. In Proc. of IMC, 2006.

[10]K. Tati and G. M. Voelker. On object maintenance in peer-to-peer systems. In Proc. IPTPS, 2006.

[11]A. Shokrollahi. Raptor codes. IEEE Trans. on Information Theory, June 2006.

[12]M. Luby. Lt codes. Proc. IEEE Foundations of Computer Science (FOCS), 2002.

[13]A.G. Dimakis, P. Godfrey et al. Network Coding for Distributed Storage Systems. In Proc of INFOCOM, 2007.

[14]S. Saroiu et al. A measurement study of peer-to-peer file sharing systems. In Proc. of MMCN, 2002.

[15] W. K. Lin, D. M. Chiu, Y. B. Lee Erasure Code Replication Revisited. In Proc. of P2P, 2004.

[16]Ananth Rao et al. Load Balancing in Structured P2P Systems. In Proc. of IPTPS, 2003.

[17]David R. Karger and Matthias Ruhl.Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems. In Proc. of IPTPS, 2004.

[18]E. Brunskill. Building peer-to-peer systems with Chord, a distributed lookup service. In Proc. of the Eighth Workshop on Hot Topics in Operating Systems, 2001.

[19]A. Ghodsi et al. Symmetric replication for structured peer-to-peer systems. In Proc. of the 3rd International Workshop on Databases, Information Systems and Peer-to-Peer Computing, 2004.

[20]Jeremy Stribling. Planetlab all pairs ping. http://infospect.planet-lab.org/pings.

[21]Sheldon M. Ross. Introduction to probability models, 8th Edition. Elsevier, 2003.

[22]Kirsten Hildrum et al. Asymptotically Efficient Approaches to Fault-Tolerance in Peer-to-Peer Networks. In Proc. of the 17th International Symposium on Distributed Computing, 2004.

[23]Jared Saia et al. Dynamically Fault-Tolerant Content Addressable Networks. In Proc. of HOT-P2P, 2004.

[24] Cyrus Harvesf and Douglas M. Blough. The Effect of Replica Placement on Routing Robustness in Distributed Hash Tables. In Proc. of P2P, 2006.

[25] Guangping Xu, Gang Wang, Jing Liu. A hybrid redundancy approach for data availability in structured P2P network systems, in Proc. of PRDC, 2007.