Route maintenance overheads in DHT overlays

Karl Aberer, Anwitaman Datta, Manfred Hauswirth
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{Karl.Aberer, Anwitaman.Datta, Manfred.Hauswirth}@epfl.ch

Abstract

Efficient route maintenance in DHTs is still an area of very active research due to its complexity and multitude of aspects to be considered. In this paper we propose novel correction-on-failure (CoF) and correction-on-use (CoU) approaches that support route maintenance more efficiently than existing methods even under highly dynamical network conditions. In contrast to previous work which addresses static resilience, we apply the more realistic model of viewing changes in the network as a continuous Markovian process and demonstrate that the system can reach a dynamic equilibrium in the presence of continuous changes while remaining operational, efficient, and scalable. We devise a generally applicable method for analyzing the dynamic behavior of route maintenance and use it to proof the efficiency of our approach. The equilibrium equations derived from the analytical model allow us to predict a system's behavior over a wide range of parameters and demonstrate its scalability. Simulation results additionally verify our analytical results. Our approach also introduces the principle of data independence into route maintenance which we demonstrate to be achievable at low cost. This separation of concern disentangles the overlay from the underlying network dynamics and is an important step towards semantic overlay networks as a basic constituent in distributed information management. It specifically facilitates the application of the P2P paradigm in mobile ad-hoc networks, identity management, tracking of past interactions (e.g., for reputation management), etc. which are of basic importance for overlay supported P2P commerce. Moreover, other domains, for example P2P based publish/subscribe systems, will also benefit from this property of our approach.

1 Introduction

Overlay networks based on distributed hash tables (DHTs), for example, [1, 12, 13, 14, 15, 17], are based on the principle of peers maintaining routing tables which facilitate the forwarding of a query "closer" to a responsible peer until the query can be answered. The ways to partition the search space and the routing methods may differ, but all DHT overlays depend on the consistent maintenance of routing tables at each peer, taking into account network dynamics, most importantly, on-/offline behavior of peers and peers' changing IP addresses.

A number of approaches have been devised which can be classified into proactive correction (PC) using periodic probing or heartbeats, e.g., [10, 17], and reactive mechanisms with the sub-categories correction on use (CoU), e.g., [1, 5], correction on failure (CoF), e.g,, [1], and correction on change (CoC), e.g., [14] .

A first contribution of this paper are novel CoF and CoU approaches that adaptively support network maintenance under various conditions of network dynamics and query frequency, that are less efficiently supported by the existing approaches. We will give a qualitative comparison of the maintenance problem space and the applicability of the different categories of approaches.

The practical usability of a DHT overlay critically depends on the efficiency of routing table maintenance. The problem boils down to maintaining a "sufficient" level of consistency while minimizing effort. Since a dynamically evolving DHT on top of a dynamically changing network is a complex dynamical system, the goal is to arrive at a stable dynamic equilibrium for a variety of conditions while guaranteeing successful routing.

Existing maintenance approaches implicitly deal with route maintenance and network dynamics as an exception, i.e., focusing on stabilization after changes, e.g., [14, 17]. Some work on static resilience, studying the performance of searches during the stabilization process has also been done [7, 9, 13]. We suggest to view changes in the network as a continuous dynamic process instead. It is more important to analyze whether the system reaches a dynamic equilibrium in the presence of continuous changes in addition to looking at recovery and stabilization issues after isolated occurrences of changes.

Thus, a second contribution is a method for analyzing the dynamic behavior of maintenance that we apply to our approach, but which is of general applicability. We model overlay network maintenance as a Markovian process and determine its equilibrium equations. By solving them we can make predictions of the system behavior for a wide range of parameters, in particular we demonstrate the scalability of the approach. Simulation experiments confirm the accuracy of the models.

A third contribution is the application of the principle of data independence [8]. Our solution disentangles the overlay structure from the dynamics of the underlying network (separation of concern). We achieve this by considering route maintenance as a subproblem of the more general identity management problem. Thus we can apply our approach not only to route maintenance but also for tracking changes in the mapping. This is important as soon as information on the characteristics of specific peers is exploited for routing purposes, such as their trustworthiness, QoS, or locality. For example, preferred routes (e.g., based on earlier observations of the routing behavior) can be maintained even if peers change their physical IP addresses or are unavailable for some periods. Approaches changing the logical IDs or restructuring routing tables as a result of changes to the peers physical ID, e.g., Chord [14], Pastry [10], would incur a loss of information. Separation of concerns is also of interest when constructing second-level overlays using a DHT as substrate (for application or optimization purposes). Then their maintenance becomes independent of the physical network dynamics.

Even though our approach is self-referential, since we use the overlay itself for keeping information used in route maintenance, we show that it is does not compromise efficiency and is robust due to a self-healing capability while being completely decentralized and self-maintaining. The maintenance effort grows gracefully with increasing network dynamics and scales logarithmically with the network size.

2 P-Grid overview

We use P-Grid [1, 4] to analyze and verify the effectiveness of our route maintenance approach. P-Grid realizes a distributed search tree that is constructed by randomized interaction of peers, and provides prefix-based search (including the capability of range queries) with $O(\log n)$ search costs. n denotes the total number of leaves in the P-Grid. Assuming an average replication factor of R, the total peer population then is $N = nR$. The prefix (or key) a peer is responsible for is called the peer's path. The association of keys to peers is performed dynamically and is independent of peer identifiers (unlike most contemporary DHTs). Multiple peers being responsible for the same leaf node form an unstructured replica subnetwork. When two peers with the same path meet, they may decide to become mutual replicas or split their key space (extend their paths by complementary bits) determined by load-balancing concerns [3]. In case of a split they reference each other at a new level in their routing tables. When peers with different paths meet, they may exchange routing references and reference each other. Unlike PRR [11] and its derivatives like Tapestry [17], P-Grid's search tree is not hierarchical (as considered to be the case for all tree structured DHTs in the DHT routing geometry study [7]). Nevertheless, in a P-Grid tree with n leaves any node has the freedom of choosing among 2^{i-1} paths at a distance i from itself, and hence has $n^{\frac{\log_2 n}{2}}$ possible routing tables. Like in the tree geometry [7] P-Grid can reduce the key-to-peer path distance by at least one in the routing process, but in contrast to the conventional understanding of the tree geometry [7], P-Grid can possibly resolve even more bits in one step due to longest prefix match [3]. In respect to routing flexibility as defined by [7], P-Grid is more flexible than many other DHTs: Given a particular choice of routing tables, if a peer receives a query with i bits unresolved, then with probability 2^{-j}, j bits can be locally resolved. Thus, apart from the one choice to resolve all these j bits, there are $j-1$ more choices, and this happens with probability 2^{-j}. Thus for i unresolved bits, on an average $1 + \sum_{j=1}^{i}(j-1)2^{-j} = 2 - (1+i)2^{-i}$ routing choices are available, giving P-Grid greater flexibility than other tree structured DHTs.

The expected number of steps is $\frac{\log_2 i}{2}$ when i bits are unresolved in a balanced P-Grid, and bounded by $\ln i$ in the case of a more general, unbalanced P-Grid [3]. Since multiple peers can have the same path in the P-Grid search tree (replica subnetwork), there is also no need of the notion of special sequential neighbors for fault tolerance. Moreover, for each depth of the tree, r routing references are cached, chosen either randomly, or by some other local criteria, e.g., proximity, which makes P-Grid even more flexible and resilient.

3 Approach

The primary functionality of any overlay is to provide a directory to efficiently index and retrieve information available at the participating peers. We use the same functionality also to consistently map a universally unique peer ID onto the peer's current IP address. The ID does not bear any semantics other than mere identification and is constructed by applying a cryptographically secure hash function to some random data. Each peer generates an ID locally once and each time it changes its physical IP address, it inserts its current ID-IP mapping into the overlay using the constant ID as the key. The routing tables hold only logical IDs which in the routing process are mapped onto physical addresses by querying a local cache for known mappings. In case no correct mapping is found in the cache, the peer queries the overlay. If a physical address is re-used by a different peer this can be detected by a simple challenge-response scheme with signatures. To prevent security attacks we apply a PGP-like security model combined with a quorum-based query scheme. To keep the overlay self-contained we store the mappings in the same overlay that depends on the mappings for route maintenance. We will show in the following that despite this recursive dependency, it is in fact possible to operate such a directory service efficiently as long as the rate of routing entries turning stale, matches the rate at which they can be repaired, i.e., we can achieve a dynamic equilibrium.

We will discuss two route maintenance strategies: With *correction on failure* (CoF) the directory is only queried if all r

routing entries at a certain level have become stale and thus further routing is impossible; with *correction on use* (CoU) a stale entry is repaired immediately by querying the directory, which is similar to the approach proposed in [5]. Of course, a query for a new mapping could again require further recursive queries due to other stale mappings. Our algorithm takes this into account with acceptable overhead as discussed in Section 4. However, a positive side-effect of recursive queries is that they implicitly lead to a self-healing effect due to the rectification of stale entries at the involved peers. A detailed presentation of the algorithms and protocols is given in [2].

An interesting observation is that recently a similar hen-and-egg problem has been shown to be feasible in the context of overlays, where a Tapestry [17] overlay has been used to transparently redirect legacy application traffic in presence of network failures [16].

4 Overheads of overlays

We provide a summary of our findings in this section. Detailed analysis and results are provided in [2].

The use of r routing references at each level of a P-Grid routing table implies storage costs of $r\log_2(N/R)$ at each peer where N is the total peer population and R is the replication factor. Additionally, in order to realize a self-contained directory, ID-IP mapping for each peer is stored in R replicas and incurs NR memory in total, and hence R memory per peer (independent of the network population). Whenever a new peer joins, or an old peer rejoins with a changed ID-IP mapping, insertion of the new mapping in P-Grid costs $O(\log_2(N/R))$ messages for locating one replica, and $O(R)$ messages for gossiping it within the replica subnetwork.

4.1 Dynamic equilibrium

Conventionally a DHT's resilience against network dynamics is measured in terms of the latency of stabilization of the overlay after such changes [14, 17], and the system's static resilience [7, 9, 13]. However, a more appropriate characteristic to study would be the overlay's behavior while in a dynamic equilibrium, because of the continuous changes and repairs triggered from network dynamics, such that the system has to continuously be self-repairing, rather than giving special treatment to the recovery phase.

In the analysis below we use the following notations: p_{on} denotes the probability of peers being online; p_{dyn} defines the probability that a randomly selected entry of a routing table is stale; ϵ_r defines the probability of failure of a recursive query; n is the number of leaves; r is the number of references for the other half of the subtree in P-Grid routing tables for each depth; and N_{rec} is the expected number of queries created per original query in the network, including the original query and its recursive children queries.

The model we analyze assumes that peers independently update their address with probability r_{up} (caused by join or rejoin) and issue queries with probability $1-r_{up}$. Since updates necessarily imply the execution of a query to locate a node at which the update is to be performed (and gossiped within its replica sub-network), we assume that $r_{up} \leq 0.5$. Thus r_{up} decouples the network dynamics from absolute time, and instead represents it with respect to network usage (queries). The average availability of peers is already accounted for with the parameter p_{on}.

The dynamic equilibrium equation for the eager strategy (CoU) then is

$$(1-r_{up})(N_{rec}-1)p_{on}(1-\epsilon_r) = r_{up}(1-p_{dyn})r\log_2 n.$$

The left hand side of the equation represents the number of routing entries repaired because of a $1-r_{up}$ fraction of queries, each of which creates $N_{rec}-1$ recursive queries on average. Each recursive query succeeds with probability $1-\epsilon_r$ and fetches the latest ID-IP mapping for a routing entry and repairs the entry provided the peer is online. During the same period, some non-stale cached routing entries may be rendered useless because of changes in the network, which is accounted for in the right hand side of the equation. At the dynamic equilibrium, the overall number of changes should equal the overall number of repairs in the P2P network. In order to solve the equations we derive further relationships among N_{rec}, ϵ_r, and p_{dyn}, which we omit here for space limitations.

For the lazy approach (CoF) recursions are triggered when i routing entries are stale, and the remaining $r-i$ are offline. Let S_i be the probability that i of the r entries in the routing table are stale, in the whole network. Thus the dynamic equilibrium equation is given by:

$$\begin{aligned}
r_{up} \quad & S_i \quad \frac{r-i}{r} r\log_2 n + (1-r_{up})\sum_{j=i+2}^{r}\frac{1}{p_{rec}}(1-p_{on})^{r-j} \\
& S_j\frac{1}{j}(N_{rec}-1)\binom{i}{j-i-1}(1-\epsilon_r)^{j-i-1}\epsilon_r^{i+1} \\
= \quad & r_{up}S_{i+1}\frac{r-i-1}{r}r\log_2 n + (1-r_{up})\frac{1}{p_{rec}}S_{i+1} \\
& (1-p_{on})^{r-i-1}\frac{1}{i+1}(N_{rec}-1)(1-\epsilon_r^{i+1})
\end{aligned}$$

The left hand side of the equation is the inflow into state S_{i+1} from S_i as well as from $S_j, \forall j > i+1$ because of partial repairs. The right hand side is the outflow from S_{i+1}. The outflow is caused by two factors: The first is because additional entries turn stale; the second occurs whenever recursions are initiated and at least one cached entry is repaired.

4.2 Results

We provide some indicative results and refer the reader to [2] for a comprehensive discussion. Figure 1 shows S_r (the probability that all r entries for any depth are stale) against r_{up} for

CoF and a balanced P-Grid for various network populations, and compares the analytical prediction with simulation observations (for R=8 and r=4, and varying N). The analytical result matches well with the simulation results once the network population is moderately high (N=256 onward), which is expected because the assumed independence property and the statistical properties derived hold only for larger network sizes.

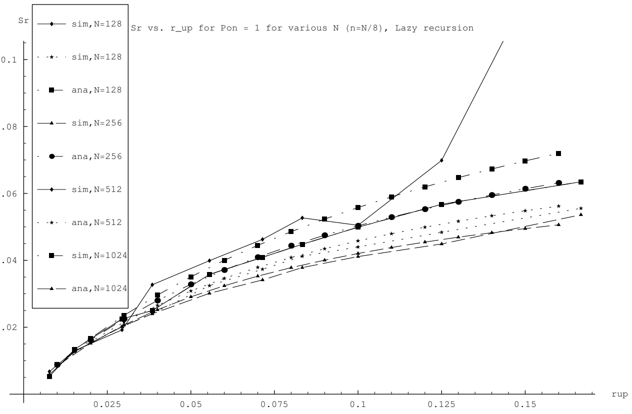

Figure 1: Correctness of analysis

Figure 2 shows the variation of overhead N_{rec} with the variation of p_{on} for various values of r_{up} for CoF. We observe that the mechanism is robust for a very broad range of p_{on} values, for any given degree of dynamics determined by r_{up}, with marginal increase in cost.

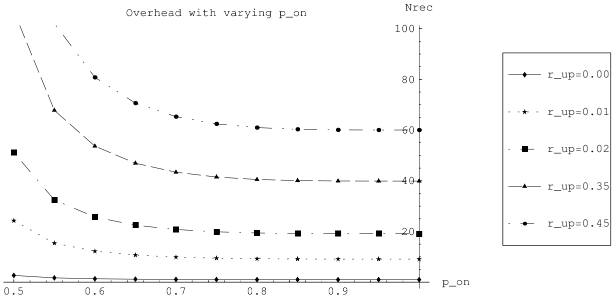

Figure 2: Overhead

Figure 3 shows the overhead in terms of messages ($\frac{N_{rec}\log_2 n}{2}$) for the lazy and eager mechanisms for a wide range of network environments with variation of p_{on} and a fixed $r_{up} = 0.2$. We observe that with very poor network reliability (low p_{on} and high r_{up}), CoU performs better than CoF. This is so because with CoF we allow unusable routing tables to accumulate, such that there is a thresholding behavior, and the network performance rapidly deteriorates beyond it. On the other hand, when network reliability is better, CoF performance is better because it does not put effort in maintenance continuously, and instead lets inconsistencies accumulate before repairing them all.

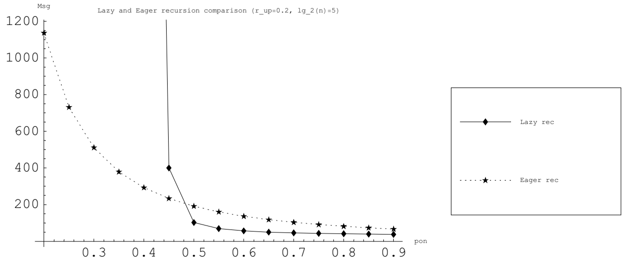

Figure 3: Eager vs. lazy

While the use of recursion (CoF and CoU) almost eliminates failures tolerating even very low p_{on} values and moderately high network dynamics (high r_{up}), the incurred effort may be too high in a real network. In Figure 4 we thus provide contour maps for CoF corresponding to N_{rec} values, with p_{on} on the X-axis and r_{up} on the Y-axis. The interpretation of the plot is that if a system (participating peers) is willing to incur an N_{rec} fold increase of effort per query with respect to the ideal case ($p_{on} = 1$ and $r_{up} = 0$), then the network will operate for all p_{on}, r_{up} combinations below the curve. If N_{rec} effort is considered too high, then the system operates above the curves with a non-zero failure probability which increases with the distance from the curve. Figure 4 thus captures two important trade-offs in the system. The first trade-off is that of efficiency versus probabilistic success guarantees of queries. The second trade-off is the system's resilience against the two "demons" of the network, the network dynamics r_{up} versus average unavailability of peers in the network $(1 - p_{on})$, which in combination determine the network reliability.

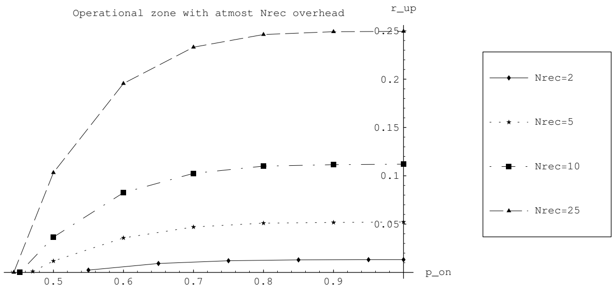

Figure 4: Contour map

4.3 Qualitative assessment of self-maintenance

The self-maintenance mechanism proposed in this paper is considerably different from other route maintenance schemes in two respects: (1) Our proposal is the first one to apply a self-contained directory and (2) we explicitly model and address two sources of unreliability of the network, the dynamics of the underlying network (r_{up}), and the average peer unavailability ($1-p_{on}$), and demonstrate that the self-contained directory may be used to heal the stale routing entries. Moreover, nodes joining, rejoining or leaving the network do not affect the structure of the network (key-to-peer associations do not change) even if query forwarding paths change and adapt to changes in the network topology.

The need for self-maintenance of routing tables is advocated and its incurred costs are evaluated in [10, 14]. The simple and effective probing mechanism of [10] is a PC mechanism which is very inefficient if there are infrequent changes. In comparison, CoC which is used in Chord (for node insertions) is a pragmatic downright reactive mechanism. CoC exploits the fact that *if there are no changes, there is no need for corrections*. CoC approaches devote their effort uniformly to one and all changes in the network. Ideally, however, the effort of maintenance should be proportional to the utility of that part of the network, which calls for a reactive mechanism like CoU which initiates maintenance only if a reference is indeed required but stale. CoF is even more pragmatic since it relies on the overlay's routing redundancy (similar to [16]) and hence resilience of the overlay, such that as long as a query may be answered, albeit with increased latency and effort (as shown in Figures 2 and 4), no repair is done. However, CoF needs some mechanism to rediscover usable routing entries, for which we can use the self-contained directory. Further study is required to see if such degree of laziness can be affordable even without the use of a self-contained directory. While CoF has the least overhead, and typically acceptable latency for a wide range of network reliability conditions, it is unsuitable in an environment where network reliability and query frequency are very low. This is because, by allowing unusable entries to accumulate by not repairing them (unlike CoU) as long as there is no failure, CoF allows the network to reach a "point of no return." Intuitively, there is a *phase transition*, and the network gets totally disconnected. Percolation theoretic analysis (similar to observations in mobile ad-hoc networks [6]) to determine the exact point of such a transition is still an open issue, and a quantitative comparison of all the existing self-maintenance mechanisms is part of our ongoing work. In the meanwhile, we provide an informal taxonomy of the overlay maintenance mechanisms in Figure 5.

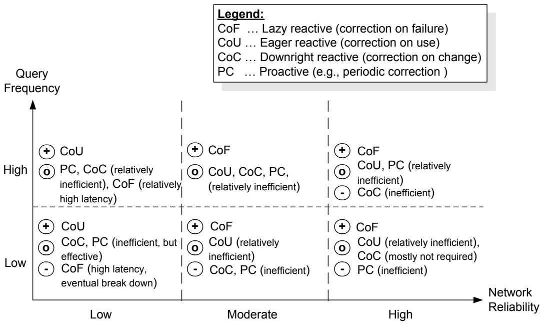
Legend:
CoF … Lazy reactive (correction on failure)
CoU … Eager reactive (correction on use)
CoC … Downright reactive (correction on change)
PC … Proactive (e.g., periodic correction)
Query Frequency
High
Low
Network Reliability
Low
Moderate
High
+ CoU
o PC, CoC (relatively inefficient), CoF (relatively high latency)
+ CoF
o CoU, CoC, PC, (relatively inefficient)
+ CoF
o CoU, PC (relatively inefficient)
- CoC (inefficient)
+ CoU
o CoC, PC (inefficient, but effective)
- CoF (high latency, eventual break down)
+ CoF
o CoU (relatively inefficient)
- CoC, PC (inefficient)
+ CoF
o CoU (relatively inefficient), CoC (mostly not required)
- PC (inefficient)

Figure 5: Applicability of maintenance mechanisms

From Figure 5 and the discussion above we conclude that none of the existing approaches is suitable for all network conditions and usage patterns. Hence hybrid mechanisms are required. The self-tuning mechanism in [10] is an example where the probing period of PC is adapted based on the change rate, thus realizing aspects of CoC. However, neither PC nor CoC exploit the fact that maintenance is needed for only what is used, and incur higher overheads. Thus hybrid CoU/CoF route maintenance mechanisms with self-tuning (when to start with repairs) are promising candidates. Such hybrid mechanisms would require only minor modification of our present algorithms and implementation. However, it is even more interesting to determine the optimal choice, which requires a detailed study of the phase transition process.

5 Conclusions

Efficient route maintenance in DHTs is still an area of very active research due to its complexity and multitude of aspects to be considered. In this paper we have proposed novel CoF and CoU approaches that support route maintenance more efficiently than existing methods even under highly dynamical network conditions. Our approach also introduces the principle of data independence into route maintenance which we demonstrate to be achievable at low cost. This separation of concern disentangles the overlay from the underlying network dynamics and is an important step towards semantic overlay networks as a basic constituent in distributed information management. It specifically facilitates the application of the P2P paradigm in mobile ad-hoc networks, identity management, tracking of past interactions (e.g., for reputation management), etc. which are of basic importance for overlay supported P2P commerce. Moreover, other domains, for example P2P based publish/subscribe systems, will also benefit from this property of our approach.

References

[1] K. Aberer. P-Grid: A self-organizing access structure for P2P information systems. In *COOPIS*, 2001.

[2] K. Aberer, A. Datta, and M. Hauswirth. Efficient, self-contained handling of identity in peer-to-peer systems. Technical Report IC/2003/36, EPFL, 2003.

[3] K. Aberer, A. Datta, and M. Hauswirth. The Quest for Balancing Peer Load in Structured Peer-to-Peer Systems. Technical Report IC/2003/32, EPFL, 2003.

[4] Karl Aberer, Manfred Hauswirth, Magdalena Punceva, and Roman Schmidt. Improving Data Access in P2P Systems. *IEEE Internet Computing*, 6(1), 2002.

[5] L. Onana Alima, S. El-Ansary, P. Brand, and S. Haridi. DKS(N,k,f): A Family of Low Communication, Scalable and Fault-Tolerant Infrastructures for P2P Applications. In *3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, 2003.

[6] O. Dousse, P. Thiran, and M. Hasler. Connectivity in ad-hoc and hybrid networks. In *IEEE Infocom*, pages 1079–1088, 2002.

[7] K. Gummadi, R. Gummadi, S. Ratnasamy, S. Shenker, and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. In *SIGCOMM*, 2003.

[8] Joseph M. Hellerstein. Toward network data independence. *SIGMOD Record*, 32(3), 2003.

[9] D. Loguinov, A. Kumar, V. Rai, and S. Ganesh. Graph-theoretic analysis of structured peer-to-peer systems: routing distances and fault resilience. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 395–406. ACM Press, 2003.

[10] R. Mahajan, M. Castro, and A. Rowstron. Controlling the Cost of Reliability in Peer-to-Peer Overlays. In *IPTPS*, 2003.

[11] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing Nearby Copies of Replicated Objects in a Distribute d Environment. In *SPAA*, 1997.

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

[13] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany*, 2001.

[14] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of ACM SIGCOMM*, 2001.

[15] Z. Xu, C. Tang, and Z. Zhang. Building topology-aware overlays using global soft-state. In *ICDCS*, 2003.

[16] B. Zhao, L. Huang, J. Stribling, A.D. Joseph, and J.D. Kubiatowicz. Exploiting routing redundancy via structured peer-to-peer overlays. In *ICNP*, 2003.

[17] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-are location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley, 2001.