The Design and Evaluation of Techniques for Route Diversity in Distributed Hash Tables [*]

Cyrus Harvesf and Douglas M. Blough
Georgia Institute of Technology
School of Electrical and Computer Engineering, Atlanta, GA 30332-0250
{charvesf,dblough}@ece.gatech.edu

Abstract

We present a replica placement scheme for any distributed hash table that uses a prefix-matching routing scheme and evaluate the number of replicas necessary to produce a desired number of disjoint routes. We show through simulation that this placement can make a significant improvement in routing robustness over other placements. Furthermore, we consider another route diversity mechanism that we call neighbor set routing and show that, when used with our replica placement, it can successfully route messages to a correct replica even with a quarter of the nodes in the system failed at random. Finally, we demonstrate a family of replica query strategies that can trade off response time and system load. We present a hybrid query strategy that keeps response time low without producing too high a load.

1. Introduction

The structured peer-to-peer architecture holds each node responsible for serving data items and correctly routing messages. Malicious nodes may abuse these responsibilities and act to tamper with the correct functioning of the system. Our solution is a replica placement scheme that can be tuned to produce a desired number of disjoint routes that can be used to circumnavigate malicious nodes.

2. Route Diversity Techniques

MaxDisjoint Replica Placement. In the following discussion, we assume that node routing tables are organized as in Pastry [4]. Furthermore, we assume that routing is performed in an identifier space of size N and that the prefix-matching occurs in digits of base B.

The simplest method for generating disjoint routes is to ensure that each route uses a different routing table entry as its first hop. Our placement, which we call the MaxDisjoint placement, guarantees this by varying the length of common prefix among the replica identifiers. The following theorem specifies the necessary replication degree to produce a desired number of disjoint routes. We omit the proof because of space limitations.

Theorem. *To produce d disjoint routes from any query node to a key k in a full distributed hash table using prefix-matching routing with base $B > 1$, the key k must be replicated at $(n+1)B^m$ locations determined by the MaxDisjoint Algorithm, where $m = \lfloor \frac{d-1}{B-1} \rfloor$ and $n = (d-1) \bmod (B-1)$.*

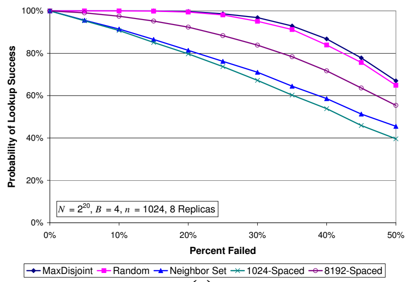
It is worth noting that disjoint routes are created without modifying the underlying routing mechanism. MaxDisjoint naturally creates disjoint routes using the prefix-matching property of the routing scheme. It can be shown rather easily that this result is consistent with the equally-spaced replica placement for Chord prescribed in [2]. The MaxDisjoint placement improves on the equally-spaced solution by allowing the replication degree to change without shifting replicas. Note that if equal-spacing is used with $B > 2$, the replication can only be doubled or halved to maintain equal-spacing without shifting any replicas. Thus, MaxDisjoint placement is a low overhead, adaptive replica placement solution.

Neighbor Set Routing. Castro et al. [1] construct a *neighbor set anycast* to route to a clustered replica set. The neighbor set anycast relies on finding a set of diverse routes from the source node to the replica set. To create route diversity, messages are routed via the neighbors of the source node. We call this technique *neighbor set routing* and measure how well it creates route diversity.

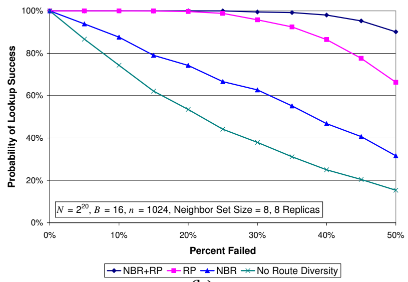
3. Experiments

For all experiments, 1024 nodes were modeled in a Pastry DHT with a 20-bit identifier space and $B = 16$ (hexadecimal digits are used in prefix-matching). For the first two experiments, each data point in the results is the average of over 100,000 lookups. Because of extended runtime, each data point in the response time experiments is the average of over 10,000 lookups.

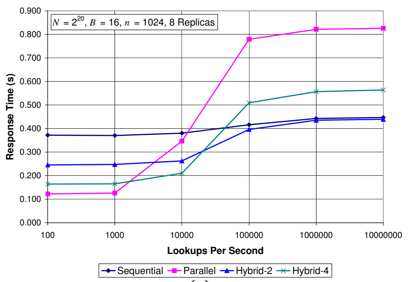
[*]This research was funded in part by the National Science Foundation under Grant ITR-NHS-0427700.



(a)



(b)



(c)

Figure 1. Experimental results: (a) replica placement, (b) combining replica placement (RP) and neighbor set routing (NBR), and (c) response time.

Replica Placement. We performed experiments to compare several replica placements in terms of the number of disjoint routes created and the impact on routing robustness. In addition to our replica placement, we also considered random placement, neighbor set placement, and spaced placement, where replicas are separated by fixed spacings of 1024 and 8192.

The impact of the placement of eight replicas on routing robustness can be seen in Figure 1(a). Our replica placement routes messages with the highest success rate over the range of failure rates tested. Even with a quarter of nodes failed, 99% of lookups are successful This is a dramatic improvement over the neighbor set placement, which only routes 75% of all messages succcessfully with a quarter of nodes failed. The random placement performs comparably to our placement, but introduces bias toward particular queries that could be exploited by an adversary.

Combining Route Diversity Techniques. To evaluate the relative impact of replica placement and neighbor set routing on routing robustness, we conducted a set of experiments to measure the routing success rate. We measured the success rate with neither route diversity mechanism, with each mechanism individually, and with both mechanisms together. The results for eight replicas and a neighbor set size of eight are shown in Figure 1(b).

There is a significant benefit in using neighbor set routing in conjunction with replica placement. Especially at higher failure rates, neighbor set routing can introduce additional diversity that can increase routing robustness. With half of the nodes in the system compromised, using neighbor set routing and replica placement together can route 90% of all lookups successfully compared to only a 66% success rate with replica placement alone.

Response Time. Finally, since replica placement seems to be a reasonable method for improving routing robustness, it is natural to consider some of the practical concerns with using replication. When querying a replica set, response time can be reduced by querying the entire replica set in parallel. However, this may have a significant impact on the system load. We consider the performance when replicas are queried in parallel, sequentially, and sequentially in sets of two or more (hybrid strategy).

We extended our fault model to assume that failed nodes correctly forward lookups to create added system load, but return incorrect responses that the query node is able to detect. Therefore, a failed route will result in the same system load as a successful route, but will add to the overall response time of the lookup. In a real system where a failure may result in no response at all, it would be necessary to use a timeout for the sequential and hybrid schemes. Correctly tuning these timeouts is beyond the scope of this paper.

In the presence of ideal conditions, the trade-off between response time and system load is clear. The parallel strategy provides less variability in response time. With increasing failure rate, the response time of the sequential strategy increases rapidly while the parallel strategy slowly increases. However, this comes at the expense of system resources. Using a hybrid strategy can exploit the trade-off to have the reduced response time variability of parallelization while reasonably controlling the system load.

To evaluate the impact of message queuing, the response time was measured for lookup rates varying from 1×10^2 to 1×10^7 lookups per second. The average response time in a system with 25% of nodes failed is shown in Figure 1(c). Clearly, the degree of parallelization can have a negative impact on the response time. Using a hybrid strategy can mitigate these effects, but the best approach is an adaptive one that varies the degree of parallelization in response to the observed system load. This is a topic for future work.

For further discussion of this work, refer to [3].

References

[1] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of OSDI '02*, pages 299–314, 2002.

[2] C. Harvesf and D. M. Blough. The effect of replica placement on routing robustness in distributed hash tables. In *Proceedings of P2P'06*, pages 57–6, 2006.

[3] C. Harvesf and D. M. Blough. The design and evaluation of techniques for route diversity in distributed hash tables. Technical Report GIT-CERCS-07-15, Georgia Institute of Technology, 2007.

[4] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Proceedings of ACM Middleware'01*, pages 329–350, 2001.