

Adaptive Routing of QoS-constrained Media Streams over Scalable Overlay Topologies

Gerald Fry and Richard West

Computer Science Department
Boston University
Boston, MA 02215
{gfry,richwest}@cs.bu.edu

Abstract

Current research in Internet-based distributed systems emphasizes the scalability of overlay topologies for efficient search and retrieval of data items, as well as routing amongst peers. However, most existing approaches fail to address the routing of data across these logical networks in accordance with quality of service (QoS) constraints. This paper investigates the use of scalable overlay topologies for the QoS-constrained routing of real-time media streams between publishers and potentially many thousands of subscribers. Our approach leverages insights gained from research on peer-to-peer (P2P) information retrieval systems, such as Chord and CAN. We analyze the costs of using similarly structured overlay networks, specifically k -ary n -cubes, for QoS-based routing. Given a number of nodes in a distributed system, we calculate the optimal k -ary n -cube structure for minimizing the average distance between any pair of nodes. Using this structure, we have developed a greedy algorithm that selects paths between nodes in accordance with the real-time delays along physical links. We show this method improves the routing latencies by as much as 67%, compared to approaches that do not consider physical link costs. Finally, we are in the process of developing a method of adapting the placement of nodes in the overlay topology based upon the locations of publishers, subscribers, physical link costs and per-subscriber QoS constraints. One such method for repositioning nodes in logical space is discussed, to improve the likelihood of meeting QoS requirements on data routed between publishers and subscribers. Future work will evaluate the benefits of such techniques more thoroughly.

1. Introduction

Recent work in the area of Internet-scale distributed systems suggests that a carefully constructed overlay topol-

ogy is beneficial for routing application-specific data. The NARADA protocol, for instance, provides strong evidence that implementing multicast functionality at the end-host level results in advantages that outweigh the delay penalties incurred over implementation in the network core [2]. Such advantages include the ability to scale to larger topologies without requiring that group state be kept at core network routers, flexibility to adapt routing behavior to application-specific events, and reliance only on unicast functionality implemented at the network layer, permitting the use of COTS-based systems on currently existing IP networks.

Although NARADA gives a convincing argument for the usefulness of end-system multicast routing, the protocol itself fails to scale as group sizes increase beyond a few hundred hosts, partly due to communication overheads introduced by random probe messages. In contrast, there have been efforts to generate more scalable overlays for storage and retrieval as well as routing of data items among peers using consistent hashing techniques. Such work includes Pastry [8], Scribe [1], CHORD [10], CAN [7] and Tapestry [11]. In summary, these systems focus mainly on scalable overlay routing without considering QoS requirements, while NARADA addresses service guarantees relating to real-time streaming media but is less scalable.

Nonetheless, it is not enough to ensure logarithmic-sized routing tables or that messages will eventually reach an appropriate destination peer. In applications where streams of multimedia data must be transmitted to a large set of subscribers with real-time constraints, it is imperative that information about the underlying physical network be leveraged in order to efficiently route the data over the logical topology. For example, consider a nationwide digital broadcast system (on the scale of Shoutcast [9]), in which hundreds of thousands of subscriber hosts receive live video feeds from one or more publishers. Such a system may require data to be delivered to each subscriber with its own unique QoS constraints.

Contributions: This work focuses on the scalable delivery of real-time media streams. We present an analysis of k -ary n -cube graphs as structures for overlay topologies [4]. In particular, an explicit formula is derived for the minimal average hop count between pairs of nodes in a k -ary n -cube graph. Using this information we develop a method for determining the optimal values for k and n to represent a logical topology involving m physical hosts. We have also devised a greedy algorithm for routing over the overlay structure while taking physical network proximity measures into account. Additionally, we investigate methods for dynamic subscriber relocation in logical space based on network proximity and per-subscriber latency constraints. Simulation results show a significant reduction in delay penalties relative to unicast delays when using the greedy routing algorithm as opposed to random dimension ordering and ordered dimensional routing.

2. Analysis of k -ary n -cube Topologies

Scalable P2P systems such as CHORD, CAN, and Pastry use distributed hashing techniques to locate objects (destination peers in the case of Pastry) in logical space. These systems require $O(\lg M)$ hops along the overlay topology, where M denotes the number of logical hosts communicating in the system [1, 10, 7]. Furthermore, the lookup services associated with these systems require that hosts maintain logarithmic sized routing tables.

We use k -ary n -cube graphs to model logical overlays for similar reasons, which for our purposes are undirected graphs specified using n as the *dimensionality* parameter and k as the *radix* in each dimension. The following properties of k -ary n -cube graphs are relevant to this work:

- $M = k^n$, where M is the number of nodes in the graph. Therefore, $n = \lg_k M$.
- Each node is of the same degree, with n neighbors if $k = 2$ or $2n$ neighbors if $k > 2$.
- The worst case shortest path between nodes in the graph is $n \lfloor \frac{k}{2} \rfloor$ hops.
- The average case shortest path between nodes in the graph is $A(k, n) = n \lfloor \frac{k^2}{4} \rfloor \frac{1}{k}$ hops.
- The optimal dimensionality of the graph is $n = \ln m$.
- Each node in the graph can be associated with a logical node identifier consisting of n coordinates, where each coordinate is an integer in base k .
- Two nodes are connected by an edge if and only if exactly one of their coordinates differ by exactly 1 (modulo k).

The regularity of k -ary n -cube graphs provides for a logical topology that is scalable in the sense that routing complexity increases less than linearly with the number of logical nodes in the system. Intuitively, the structure is regular and compact, and different choices of values for k and n

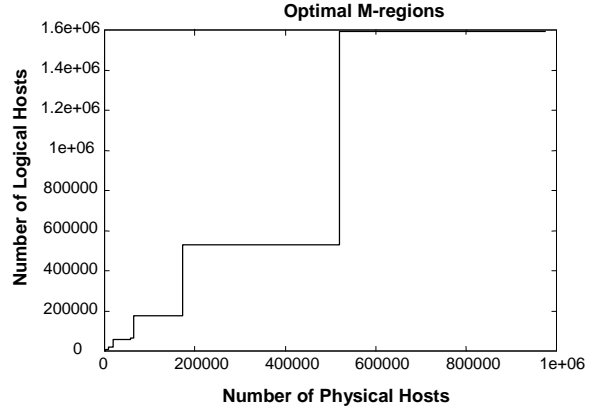


Figure 1. M-regions for $m \leq 1000000$

result in differing topology sizes and corresponding values of $A(k, n)$.

Given that m physical hosts are participating in the system, values for k and n can be found which result in a k -ary n -cube graph that is optimal with respect to minimal average shortest path hop count between pairs of nodes while simultaneously maximizing the value of $M = k^n$. The problem of choosing such k and n reduces to imposing a linear ordering on (k, n) pairs such that corresponding values of $A(k, n)$ are monotonically increasing. Thus, for each pair (k, n) we define an M -region, or a range of values for the number of physical hosts for which the associated k -ary n -cube graph is optimal.

An iterative algorithm is used to search the space of values for k and n that result in graphs with lower average hop count between nodes or a larger value for k^n with an equal average hop count as compared with the pair of k and n used as the starting point (all starting points for which $k = 2$ are used). The results are summarized in Figure 1, which plots the number of logical hosts resulting from optimal k and n values found for a given number of physical hosts.

When hosts join or depart from the system, the value of m may change, and the overlay can be adjusted to a more efficient configuration of parameters based on calculated M -regions. The advantage of this scheme is in the reduction of the average logical hop count between nodes without sacrificing scalability of the overlay topology. The routing tables associated with various pairs of values for k and n can either be computed when the size of the system changes, or be pre-calculated and stored at each host. This is similar to the concept of *realities* in CAN [7].

3. Proximity-based Greedy Routing

The effectiveness of a distributed system for real-time streaming media is heavily influenced by the way in which

data is routed throughout the network. This work investigates the performance of three algorithms used for routing over k -ary n -cube logical topologies, built on top of a physical network:

- **Ordered Dimensional Routing:**

A message is first routed towards the node which matches the destination identifier in the first coordinate of the logical node ID. Subsequently, the message passes along the second dimension until the second coordinate matches. This progresses in each dimension, in order, until the message reaches the peer with the closest matching identifier to the specified destination ID. This is the method for routing used by systems based on Pastry, such as Scribe and PeerCQ [1, 6].

- **Random Ordering of Dimensions:**

At each hop, a coordinate is chosen at random among those coordinates for which the current node’s logical ID is not equal to the destination ID. The message is forwarded to the node with a corresponding coordinate that is numerically closer to the same coordinate of the destination.

- **Greedy Routing:**

As a main contribution of this work, greedy routing is performed using some measure of physical proximity. It is assumed that each host maintains a measured cost (i.e., latency) to each of its direct neighbors in the k -ary n -cube. A message is forwarded to the neighbor along the logical edge which results in the lowest cost among all other neighbors for which forwarding reduces the distance to the destination node. Since there are $n \lfloor \frac{k^2}{4} \rfloor \frac{1}{k}$ hops on average along the overlay network between two hosts, and finding the next hop requires searching $O(n)$ neighbors, the resulting complexity of the greedy algorithm is $O(n^2k)$.

Experimental analysis was done via a simulation written in C, while leveraging gt-itm for generating random transit-stub physical topologies [3]. The physical topology contains 5,050 routers, and the system is comprised of 65,536 hosts each randomly assigned to a router. The experiment proceeds by choosing one host at random to be a publisher, and all other hosts are assumed to be subscribers. A message is then routed from the publisher host to each subscriber host and end-to-end latencies are recorded, as well as the unicast latency of a message routed directly between the publisher and each subscriber (as if the hosts are logically directly connected). The delay penalty of routing over the overlay relative to the unicast (IP layer) delay is calculated as the logical end-to-end latency divided by the unicast delay for each subscriber host.

Figure 2 shows the cumulative distribution of delay penalties for the three algorithms using two different con-

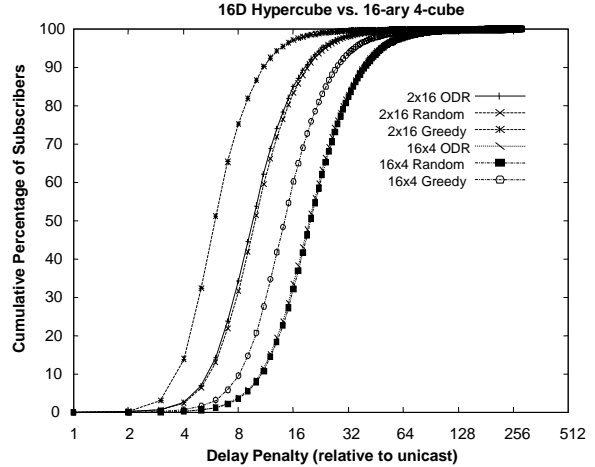


Figure 2. Comparison of routing algorithms

figurations of k and n . The values on the y -axis represent the percentage of subscribers which incur a delay penalty no more than the corresponding value on the x -axis. Simulation results indicate a significant improvement in delay penalty for greedy routing compared with random and ordered dimensional routing for both structures, whereas ordered dimensional routing performs no better than in the random case. We also see that the greedy algorithm performs better relative to the other routing methods when the node degree is greater, since this gives a higher probability of finding next hops with closer proximity in the underlying physical network. Additionally, the results show that the topology in which $k = 2$ performs better than in the case where $k = 16$, which is consistent with the analysis of M -regions in the previous section. As can be seen from Figure 2, there is as much as a 67% reduction in the relative delay penalty when using the greedy algorithm compared to the ODR or random approaches. This difference in performance is most noticeable when $k = 2$ and $n = 16$ for the 80th cumulative percentile delay penalty value.

4. Adaptive Node ID Assignment

The preceding sections focus on problems related to the initial choice of an overlay structure and methods for routing data among peers. Consider a set of subscriber hosts, each with potentially independent QoS constraints on a multimedia stream being multicast from a publisher over a k -ary n -cube logical topology. As hosts subscribe to the data stream, the information regarding quality of service can be leveraged to further adapt the position of subscribers in *logical* space by swapping logical node identifiers and any other pertinent routing state. We are currently investigating algorithms that swap the positions of joining subscribers with other hosts in order to increase the likelihood of sat-

isfying QoS constraints as well as to decrease the average lateness with respect to deadlines [5]. One such algorithm works as shown in Figure 3. This algorithm takes two arguments. S represents the new subscriber which is assumed to advertise its interest in receiving a data stream from the publisher host P . The notation $i.cost(P)$ denotes the total end-to-end cost of routing a message between host i along the logical topology to host P .

```

Subscribe(Subscriber S, Publisher P)

  Find the neighbor i of P such that
  i.cost(P) < S.constraint or
  i.cost(P) is minimum for all neighbors

  If host i is not a subscriber
  then swap logical positions of i and S

  If host i is a subscriber
  then Subscribe(S, i)

```

Figure 3. Adaptive node re-assignment algorithm

The algorithm recursively checks for positions appropriate for reassignment of subscribers in the overlay starting from the publisher node. Experiments are being conducted to investigate the effect of this logical node placement on QoS success rate and lateness. Other algorithms currently under investigation include a first-fit latency-constrained algorithm where subscribers can be displaced from their positions by newer subscribers for various constant bounds on the number of times a node can be displaced and a different algorithm that only allows swapping between those pairs of nodes which will result in an increase in a utility function associated with QoS constraints and physical network proximity.

5. Conclusions and Future Work

This work analyzes the use of k -ary n -cubes for routing real-time media streams between publishers and potentially hundreds of thousands of subscribers, in keeping with per-subscriber service constraints. We analyze the minimal average hop-count between any pair of nodes in a k -ary n -cube and use this as the basis for constructing an overlay topology for real-time transport of data. This work extends the concept of *realities*, first described in the context of CAN [7], to determine M -regions. These are regions describing, for a given number of physical hosts in a system (m) the optimal values for k and n in the corresponding overlay structure. Using our greedy algorithm, which leverages physical proximity information, we are able to route

over such topologies with significantly lower delay penalties than existing approaches based on peer-to-peer routing.

Future work includes further analysis and simulation of the algorithms outlined for adaptive reassignment of subscriber nodes in logical space and investigation into how changing the overlay structure affects per-subscriber QoS constraints for real-time media streams. The complexity of the algorithm defined in Figure 3 will be derived, and we also plan to investigate multicast algorithms involving proxying of information at multiple hosts in order to more efficiently distribute data to subscribers. Our goal is to build an adaptive distributed system capable of providing the QoS guarantees of NARADA while maintaining the scalability of systems such as Pastry/Scribe.

References

- [1] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications (JSAC)*, 2002. To appear.
- [2] Y.-H. Chu, S. G. Rao, and H. Zhang. A case for end system multicast. In *ACM SIGMETRICS 2000*, pages 1–12, Santa Clara, CA, June 2000. ACM.
- [3] K. C. Ellen Zegura and S. Bhattacharjee. How to model an internetwork.
- [4] M. K. et al. Isomorphic strategy for processor allocation in k -ary n -cube systems. *IEEE Transactions on Computers*, Vol. 52, No. 5, pages 645–657, May 2003.
- [5] G. Fry and R. West. Adaptive routing of qos-constrained media streams over scalable overlay topologies (tech report currently under development). Technical report, Computer Science Department, Boston University.
- [6] B. Gedik and L. Liu. Peercq: A decentralized and self-configuring peer-to-peer information monitoring system. In *ICDCS 2003*, 2003.
- [7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 161–172. ACM Press, 2001.
- [8] A. Rowstron and P. Druschel. 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)*, pages 329–350, Heidelberg, Germany, November 2001.
- [9] Shoutcast: <http://www.shoutcast.com>.
- [10] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.
- [11] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division, U. C. Berkeley, Apr. 2001.