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 on 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 transport of data across these logical networks in accordance with quality of service (QoS) constraints. Consequently, this paper investigates the use of scalable overlay topologies for routing real-time media streams between publishers and potentially many thousands of subscribers. Specifically, we analyze the costs of using k-ary n-cubes for QoS-constrained 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 describe 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* 40%*, compared to approaches that do not consider physical link costs.*

We are in the process of developing a method for adaptive node placement 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 service 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 topology 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: (1) the ability to scale to larger topologies without requiring that group state be kept at core network routers, (2) flexibility to adapt routing behavior to application-specific events, and (3) reliance only on unicast functionality implemented at the network layer, permitting the use of COTS-based systems on 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]. However, unlike NARADA, these systems make no explicit attempt to route data in accordance with latency and bandwidth requirements.

For real-time routing, it is not enough to use scalable overlays such as those described above. 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. In the absence of information about physical “proximities” between nodes, data could be routed over links that have large latencies or low bandwidths.

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 [6]. In particular, we develop a method for determining the optimal values of k and n, to represent a logical topology supporting m physical hosts. We describe 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 and ordered dimensional routing.

2. Analysis of *k-ary n-cube* Topologies

Scalable peer-to-peer (P2P) systems such as CHORD, CAN and Pastry use distributed hashing techniques for locating objects (and nodes) in logical space. These systems route in as little as $O(\lg M)$ 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 up to $O(\lg M)$ sized routing tables.

We use undirected *k-ary n-cube* graphs to model logical overlays in a similar manner to the P2P systems described above. These graphs are specified using n as the *dimensionality* parameter and k as the *radix* (or *base*) 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 minimum distance between any pair of nodes in the graph is no more than $n\lfloor \frac{k}{2} \rfloor$ hops.
- The average routing path length 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 identifier consisting of n digits, where the ith digit (given $1{\le}i{\le}n$) is a base-k integer representing the offset in dimension i.
- Two nodes are connected by an edge *iff* their identifiers have $n - 1$ identical digits, except for the ith digit in both identifiers, which differ by exactly 1 modulo k.

There is not necessarily a one-to-one mapping between *physical* hosts and nodes in the *k-ary n-cube* graph representing the overlay network. However, for such a logical structure to be useful for routing, we require that the number of *physical* hosts, m, be less than or equal to the number of *k-ary n-cube* nodes, M, representing the *logical* hosts. The case in which $m < M$ requires that some *physical* hosts be responsible for performing the routing functions of multiple *logical* nodes, including maintenance of the corresponding routing tables and proximities of immediate neighbors in the overlay topology.

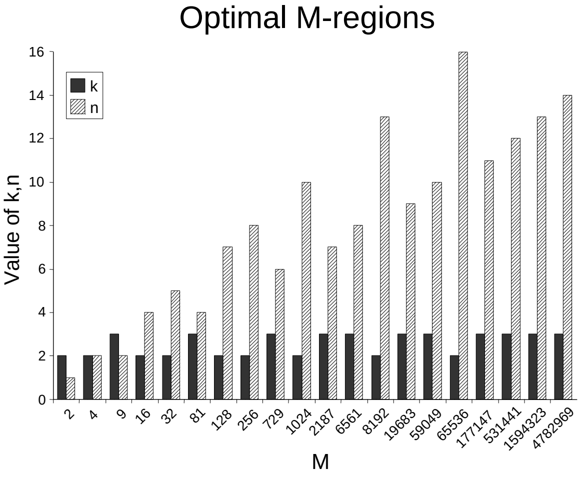

Figure 1. Graph of M-regions

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, with different values of k and n resulting 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 for a *k-ary n-cube* that is optimal with respect to hop count between pairs of nodes, while simultaneously maximizing the value of $M = k^n$. The problem of choosing values for k and n reduces to imposing a linear ordering on (k, n) pairs such that corresponding values of $A(k, n)$ are monotonically increasing. Further details can be found in an accompanying Technical Report [4]. For each (k, n) pair we define an *M-region*, or range of values for the number of *physical* hosts, for which the associated *k-ary n-cube* is optimal with respect to $A(k, n)$.

Figure 1 shows a bar chart of the first twenty *M-regions*, with M on the horizontal axis and values for k and n on the vertical axis. Columns for k and n are shown side-by-side for each size of the logical network corresponding to the appropriate *M-region*. For example, if the number of physical hosts, m, is 65000, the optimal values of k an n would be 2 and 16 respectively, such that the number of logical nodes, M, is 65536.

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

For the purposes of QoS-constrained routing, this work investigates the performance of three algorithms that leverage *k-ary n-cube* logical topologies, built on top of a physical network:

- *Ordered Dimensional Routing*: For a destination identifier, $d_1 d_2 \cdots d_n$, a message is initially routed to a node that matches d_1 in the first digit of its logical node ID. For each dimension $i \mid 1 \leq i \leq n$, the message passes to a node whose ith digit of its ID matches d_i. This is the method for routing used by systems based on Pastry, such as Scribe and PeerCQ [1, 5].
- *Random Ordering of Dimensions*: This is similar to *ordered dimensional routing* except messages are forwarded along randomly selected dimensions towards the destination. We make sure that messages are always routed closer to the destination at each hop.
- *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 $O(n^2 k)$.

Experimental Analysis: 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.

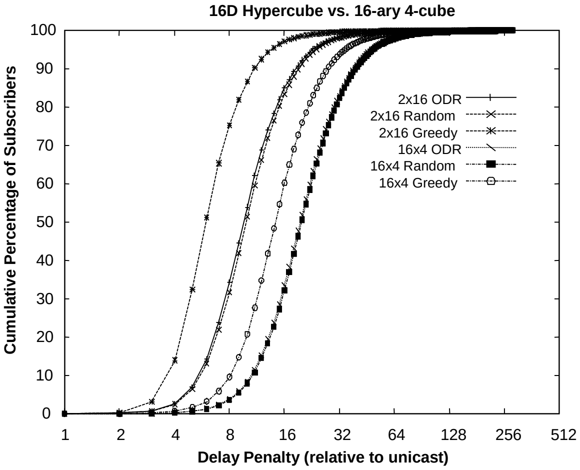

Figure 2. Comparison of routing algorithms

Figure 2 shows the cumulative distribution of delay penalties for the three algorithms using two different configurations 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 40% 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

Bootstrapping an overlay topology and randomly assigning node IDs can result in poor proximity between neighboring nodes. As a result, it is sometimes beneficial to adapt the positions and, hence, IDs of nodes in the overlay. Initially, all hosts function equally as routing agents forwarding messages across the logical topology. Once a host receives a node identifier corresponding to a position in the logical network, it can request to become a *publisher* of a new data stream or a *subscriber* to an already existing data stream. Such requests may take the form of messages routed over the optimal *k-ary n-cube* structure using the greedy algorithm described in the previous section.

As hosts begin to specify interest in receiving particular data streams with service constraints, it becomes possible to re-assign such hosts to more appropriate locations in logical space. Re-assignment of a host to a new location in the *k-ary n-cube* overlay based on the requested QoS constraints is accomplished by *swapping* the logical node identifier, as well as routing table information, with some other host in the system. We investigate an algorithm that swaps the positions of joining subscribers with other hosts in order to increase the likelihood of satisfying QoS constraints as well as to decrease the average lateness with respect to deadlines. One such algorithm works as shown in Figure 4. 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. Preliminary results show an increase of approximately 20% in overall success ratio when using this algorithm compared with the case when there is no adaptation of the overlay structure.

```
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

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 4 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] G. Fry and R. West. Adaptive routing of qos-constrained media streams over scalable overlay topologies. Technical Report BUCS-TR-2003-020, Boston University, 2003.

[5] B. Gedik and L. Liu. PeerCQ: A decentralized and self-configuring peer-to-peer information monitoring system. In *ICDCS 2003*, 2003.

[6] M. Kang, C. Yu, H. Y. Youn, B. Lee, and M. Kim. Isomorphic strategy for processor allocation in k-ary n-cube systems. *IEEE Transactions on Computers, Vol. 52, No. 5*, pages 645–657, May 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.