Turning Heterogeneity into an Advantage in Overlay Routing

Zhichen Xu
Hewlett-Packard Laboratories
1501 Page Mill Rd
Palo Alto, CA 94304
Email: zhichen@hpl.hp.com

Mallik Mahalingam
VMware Inc.
3145 Porter Drive
Palo Alto CA 94304
Email: mallik@vmware.com

Magnus Karlsson
Hewlett-Packard Laboratories
1501 Page Mill Rd
Palo Alto, CA 94304
Email: karlsson@hpl.hp.com

***Abstract*— Distributed hash table (DHT)-based overlay networks, represented by Pastry, CAN, and Chord, offer an administration-free and fault-tolerant application-level overlay network. While elegant from a theoretical perspective, these systems have some disadvantages. First, they rely on application-level routing, which may be inefficient with respect to network delays and bandwidth consumption. Second, they typically construct a homogeneously structured overlay even though nodes in these networks usually have varying physical connectivity and packet-forwarding capacities.**

In this paper, we propose two approaches for constructing an auxiliary *expressway* network to take advantage of the different connectivity, forwarding capacities, and availabilities of the nodes. As a result, we are able to reconcile the conflict of presenting the applications with a homogeneous structured overlay to simplify management, while at the same time taking advantage of the inherent heterogeneity of the underlying physical network to speed up routing. Our simulation results show that our expressway can achieve close to optimal routing performance (on average, 1.07 and 1.41 times optimal routing for an Internet-like topology and a large synthesized transit-stub graph, respectively) in overlay networks.

I. Introduction

Peer-to-peer (P2P) systems are gaining popularity quickly due to their scalability, fault-tolerance, and self-organizing nature. Progress in P2P has been made in applications such as storage [1], [2], DNS, media streaming [3], collaborative Web server [4], distributed content-based search [5], and even distributed firewalls [6].

Distributed hash table (DHT)-based systems such as CAN [7], Chord [8], and Pastry [9] present applications with a homogeneously structured application-level overlay network. Nodes in these networks collectively contribute towards an administration-free storage space. Retrieving an object in these systems amounts to routing to the node that is responsible for storing that object. Providing a simple and homogeneous abstraction has several desirable properties. It provides stability by hiding the underlying dynamism and heterogeneity of the system, and simplifies the management of the large-scale distributed system by e.g., providing a uniform storage space to avoid hot-spots.

While elegant from a theoretical perspective, these systems have two limitations. First, they rely on application-level routing that to a great extent ignores the characteristics of the underlying physical networks, which can lead to excessive routing delays. Second, they construct a homogeneous structured overlay network, while in reality, the nodes usually have different capacities and constraints such as *load, packet-forwarding capacities, network connections, and availability*. In fact, for different nodes, the number of nodes that are physically close to them in network distance can vary significantly. We support this claim by constructing an Autonomous System (AS) graph from BGP routing tables [10], that shows that the fan-out of an AS can range from 1 to 2621. We argue that for a system to function efficiently, it is important to make discriminative use of the nodes in the system. The question we try to answer is how we can take advantage of the heterogeneity of nodes to improve routing performance without altering the abstraction presented to the application.

In this paper, we describe two approaches for constructing an auxiliary network called an *expressway*, that takes physical proximity, forwarding capacity, node availability, and node connectivity into account, to significantly increase routing performance. The first approach uses the AS-level topology extracted from BGP reports, and the second approach uses a novel *landmark numbering* strategy that can deal with changing network conditions.

Systems such as Pastry [9] and Tapestry [11] account for physical proximity when an overlay is constructed. They assume the topology satisfies the triangle inequality, and rely on the ability to find the physically closest node at node join. Savage *et al.*[12] have shown that triangle inequality may not hold in Internet topology. Topologically-Aware CAN [13] uses landmark ordering to cluster nodes that are physically close into logical vicinity during overlay construction. Though this yields good performance improvements, it may cause significant imbalances in the distribution of the nodes in the CAN space that leads to hot-spots [14] and does not handle changing network conditions. Chord, on the other hand, does not consider physical topology when constructing the overlay. Instead, a message is forwarded to the topologically closest node among the next hop candidates in the routing table. The choices for each routing hop, unfortunately, are limited to entries in the routing table. All the above systems are constrained by the logical structure of the default overlay, possibly limiting the maximum performance that can be achieved.

Brocade [15] removes some of the constraints in previous systems by constructing a secondary overlay network of supernodes that are situated near the network access points such as routers. Though Brocade improves performance, it still uses logical routing in the secondary network, which incurs several physical hops for every logical hop. Brocade, to some extent, pushes the problem to an auxiliary network of a smaller size.

Our proposal decouples the homogeneous overlay abstraction from routing altogether. It allows nodes to have a variable number of neighbors in an expressway, leaving the homogeneous structure of the default overlay network intact. In an expressway, nodes that are situated near gateways or routers, have good fan-outs, have good forwarding capacity and/or are highly available establish connections with each other in a way that preserves physical proximity. These nodes collectively form an expressway that is used to improve routing performance by propagating route information. To our knowledge, our expressway is the first unconstrained auxiliary network that takes full advantage of heterogeneity and proximity in the underlying network.

The contributions of this paper are:

- Constructing an auxiliary network based on network proximity information using AS-level topology derived from BGP reports and a novel *landmark numbering* technique to enable proximity neighbor selection and that can handle changing network conditions.
- Route advertisement using a variant of the distance vector algorithm [16]. In particular, we employ a route summarization technique to reduce routing state while trading off routing performance. This is equivalent to advertising network prefixes in BGP with the exception that the nodes that are aggregated are logically close to each other rather than physically close to each other.
- A simulation study using an Internet-like topology and a transit-stub graph produced using GT-ITM [17]. To show the effectiveness of our techniques, we compare our approach with eCAN [18] and Brocade. eCAN is a hierarchical version of CAN that employs proximity neighbor selection to fit the structure of the overlay to that of the physical network (but is still constrained by the logical structure of the overlay). Brocade builds a secondary overlay network of supernodes.

Our simulation results show that taking underlying physical network characteristics into consideration for routing in the overlay network yields significant performance improvement over the default routing algorithm. Our approach achieves close to optimal, shortest-path routing performance. In most cases, the previous approaches Brocade and eCAN stay within 2 to 6 times the performance of optimal routing. Brocade approaches the performance of optimal routing only when address are advertised everywhere. Even then, we expect expressway routing to outperform systems like Brocade in terms of multicast performance because expressway removes the constraints imposed by the logical structure of the overlay, and better approximates the underlying physical network. In fact, constructing an overlay that closely approximates the physical network makes it possible to deliver routing performance that is better than default IP routing. This has been shown by RON [19] and Detour [20].

The remainder of the paper is organized as follows. Section II provides background on CAN and eCAN that we use in our study. Section III describes our two ways to construct expressways and their protocols. In Section IV we evaluate the approaches using simulation. Section V provides related work and Section VI concludes the paper.

II. Default Overlay Networks

Though we will explain our approach in combination with CAN [7], our approach is generic and applicable to many overlay systems such as Pastry [9] and Chord [8]. We describe CAN and an improvement of it called eCAN [18] below.

CAN abstracts the problem of data placement and retrieval over large-scale storage systems as hashing that maps “keys” onto “values”. It organizes the logical space as a d-dimensional Cartesian space (a d-torus). The Cartesian space is partitioned into zones, with one or more nodes serving as owner(s) of the zone. A key is a point in the space, and the node that owns the zone that contains the point stores the corresponding value (or object). Routing from a source node to a destination node boils down to sending a package from one zone to another neighboring zone in the Cartesian space. This is repeated until the destination is reached. Node addition corresponds to picking a random point in the Cartesian space, routing to the zone that contains the point, and splitting the zone with its current owner(s). Node removal amounts to having the owner(s) of one of the neighboring zones take over the zone owned by the departing node. In CAN, two zones are neighbors if they overlap in all but one dimension along which they neighbor each other.

eCAN augments CAN’s routing capacity with routing tables of larger span. Every k CAN zones represent an order-2 zone, and k order-i zones represents an order-$i+1$ zone. The variable k is called the zone coverage factor of eCAN. A node is an owner of a CAN zone and is also a resident of the high-order zones that encompass that CAN zone. Besides its default routing neighbors that are CAN zones, a node also has *high-order routing neighbors* that are representatives of its neighbors in the high-order zones. eCAN provides flexibility in selecting the high-order neighbors. When selecting representatives for a high-order neighbor, we can select the node that is closest to the current node among all the nodes that belong to the neighboring high-order zone. As a result, the physical routing latencies are reduced as opposed to random routing neighbor selection.

When selecting the next hop for routing to a specific destination, we employ a scheme that try to balance the physical and logical distance with an attempt to reduce the overall routing delay. That is, in stead of selecting the next hop that is closest to the current node in network distance, we select the node that is physically close to the current node and is also logically close to the destination with a hope of

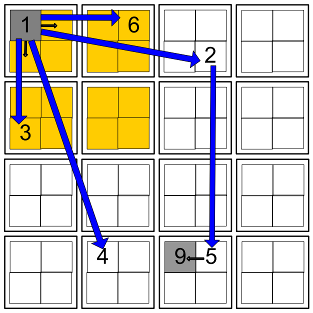

Fig. 1. An example of eCAN routing.

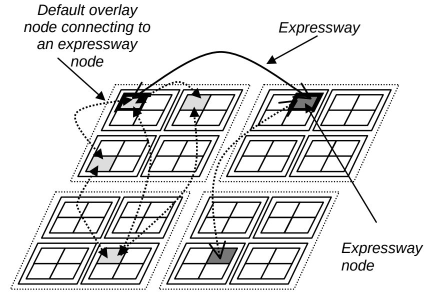

Fig. 2. Selection of expressway neighbors using AS topology.

reducing the actual routing delay. For details of this heuristic please refer to the report by Xu and Zhang [18]

Figure 1 illustrates eCAN with an example. The default CAN zones are order-1, and each of the CAN zones are 1/64 of the entire Cartesian space. In this example, four neighboring CAN zones make one order-2 eCAN zone and four order-2 zones make an order-3 zone. For example, node 1 owns a CAN zone (the zone with dark shading in the upper- left corner), and it is also a resident of the order-2 and order-3 eCAN zones that enclose the CAN zone. The routing table of node 1 consists of the default routing table of CAN (represented by the thin arrows) that link only to node 1's immediate CAN neighbors, and the high-order routing tables (represented by the thick arrows) that link to one node in each of node 1's neighboring eCAN zones at order-2 and order-3. Figure 1 also shows how node 1 can reach node 9 using eCAN routing (1 – 2 – 5 – 9).

III. Topology-Aware Expressway

The main idea of the expressway is that each node establishes connections with nodes in its physical proximity that are situated near network access points such as gateways or routers, that are highly available, and that have good fan-outs and forwarding capacities. These nodes are called *expressway nodes*. The expressway nodes themselves are linked to other expressway nodes that are nearby, called *expressway neighbors*, to form the *expressway*. We use a *distance-vector-based* route advertisement [16] in the expressway and employ a summarization technique to reduce the routing state. In our work, expressway nodes serve three purposes: (i) propagating routing information when nodes join or leave, or the network condition changes; (ii) resolving the routing destinations; (iii) forwarding packets for multicasting or when expressway routing can perform better than default IP routing.

In the subsections that follow, we describe how expressway neighbors are selected, how routing advertisement among expressway nodes is performed, and the routing algorithm.

A. Selection of Expressway Neighbors

We use two different approaches for selecting expressway neighbors. The first approach is based on deriving AS-level topology information and clustering nodes that are part of the same AS. While this information is fairly simple to derive, there are a few issues with respect to the way the AS-level graph is used for selecting expressway neighbors. First, the distance information provided by the AS graph can be too coarse grained as it is only specified in AS-level hops. Second, the AS graph captures only practically static information. Third, there can be many AS that do not participate in the overlay. Consequently, an expressway may not be fully connected. To address these problems, we propose an alternative technique, which is based on clustering nodes that observe the same latency behavior with a few chosen landmarks. Our landmark cluster scheme differs from prior work in that (1) we place the information of the nodes using landmark vectors (produced by landmark clustering) as keys on the default overlay network itself such that information about nodes that are close to each other in the physical network are placed logically close to each other on the overlay, and (2) we use Hilbert curves [21] to resolve the dimensionality mismatch between the landmark vectors and the overlay while preserving proximity relationship among nodes. We describe the details of these two approaches in the following sections.

1) Using AS Topology: When a node joins an expressway, it determines what AS it belongs to by mapping an IP address to an AS-ID using BGP reports. It then publishes this information that includes its IP address, its logical ID (e.g., node ID in the case of Chord and Tapestry, and the CAN zone a node owns in the case of CAN), and whether it is an expressway node or not. This published information is kept in the overlay system itself using a hash of the publishing node's AS-ID as the key. Every single node belonging to that AS will add its information to the previously published data. The global state published on the default overlay does not have to be kept consistent because it will not affect the correctness of the routing, but only the routing performance.

An *ordinary* (non-expressway) node uses the ID of its AS to get the information on all other nodes in the same AS, and establishes a direct connection with the local expressway node in the same AS by adding it as its expressway neighbor. To avoid having to go through the local expressway node every time a pair of nodes in the same vicinity communicates with each other, the first time an ordinary node tries to communicate with a local ordinary node, it will always go through the expressway node and cache the address of the destination. It will communicate to the destination directly from then on.

If the new node decides to act as an expressway node (i.e., finds that it has a good fan-out, have good forward capacity, and is highly available), it also uses the IDs of its neighboring AS (that it obtains from the AS-level topology) as hash keys, to obtain lists of expressway nodes in the overlay that are its topological neighbors and to establish direct connections with them. When a node joins or leaves the system, its expressway neighbors are notified in the case of a voluntary departure.

Figure 2 illustrates how expressway neighbors are selected, using eCAN as an example. The nodes that are in the light gray zones belong to the same AS and the nodes that are in the dark gray zones belong to a neighboring AS. The nodes with thick borders are expressway nodes. The expressway nodes establish connections between themselves to form an expressway. Ordinary nodes establish connections with expressway nodes in their AS. In addition, each node in an AS establishes connections with other nodes in its AS.

2) Using Landmark Clustering: In this algorithm, we pick n landmark nodes randomly scattered in the Internet. These landmark nodes can be part of the overlay itself or stand-alone. Each expressway node measures its network distance to the n landmarks. For node A, suppose that the measured distances are $< l_1, l_2, ..., l_n >$. Node A is then positioned in an n-dimensional Cartesian space using $< l_1, l_2, ..., l_n >$ as its coordinates. We call these coordinates the *landmark vector*, and this Cartesian space the *landmark space* and use it for representing nodes closeness in terms of network distance. The intuition behind doing this is that nodes that are close to each other have similar landmark measurements, and are close to each other in the landmark space.

It should be pointed out that a sufficient number of landmarks need to be chosen to reduce the probability of false clustering where nodes that are far away in network distance are clustered close to each other. To eliminate false clustering, we use landmark clustering only as a pre-selection process to identify nodes that are potentially close to a given node, and use actual round-trip time (RTT) measurements to identify the node that is the closest [22].

The expressway nodes independently determine their positions in the landmark space and publish their positions on the default overlay. An ordinary node or an expressway node finds expressway nodes that are physically close to it by referring to this published information by using its own landmark vector as the DHT-key.

The difficulty in storing the position information in the landmark space is that the landmark space is of relatively high dimension, whereas the overlay itself is of relatively low dimension (e.g., 2). To solve this problem, several techniques can be used: space-filling curves, and extending the notion of *multiple reality* introduced by CAN.

Using space-filling curves: Space-filling curves map points in the domain $\Re^1$ (the domain of real numbers) into $\Re^d$ (a d-dimension Cartesian space) so that the closeness relationships among the points are preserved. If two points are close to each other in $\Re^1$, they will also be close to each other in $\Re^d$. One example of space-filling curves is the Hilbert Curve [21]. The Hilbert curve is defined recursively. For an approximation level equal to 1 it is a point. For an approximation level equal to 3, it looks similar to Figure 3b. For each higher approximation level, we subdivide the entire space into four sub-zones and copy a shrunken and possibly rotated version of the current approximation into each sub-zone.

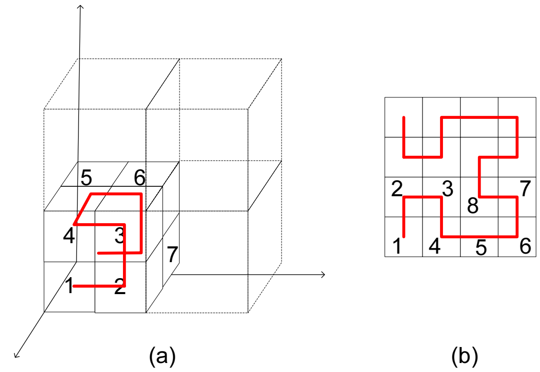

Fig. 3. Mapping a 3-dimensional landmark space (a) to 2 dimensional CAN space (b) using the Hilbert curve.

We partition the landmark space into 2^{nx} grids of equal size where n refers to number of landmarks and x controls the number of grids used to partition the landmark space. We fit a Hilbert curve onto the landmark space to number each grid. Each node inherits the grid number in which its landmark vector falls into, and we call this number the *landmark number* of the node. Closeness in landmark number indicates physical closeness. The smaller the x, the larger the likelihood that two nodes will have the same numbering, and the coarser grain the physical proximity information.

For CAN, we can partition the overlay into grids, and store the information about expressway nodes in a grid depending on its landmark numbering, using a space-filling curve (see Figure 3). We can employ a similar procedure for other overlay networks. For example, in the case of Chord, we can simply use the landmark number as the key to store the information of an expressway node on a node whose ID is equal to or greater than the landmark number. In the case of Tapestry, we can use a prefix of the node IDs to partition the logical space into grids. In summary, our goal is to store expressway node information such that information about close-by nodes is stored close to each other in the overlay.

Using multiple realities: Another approach is to construct a high-dimensional CAN solely for the purpose of storing the positions of the expressway nodes in the landmark space. Because the number of expressway nodes in the system is much smaller than the total number of nodes in the system, we can select a small number of nodes for the high-dimension CAN space. These nodes have multiple realities: the default overlay, which can be Pastry, Tapestry, Chord, and other types of overlay; and the CAN space for storing landmark information.

There exist some trade-offs between the two proposed approaches. (1) Hilbert curves do not produce any additional maintenance overhead. But, more false clustering will proba-

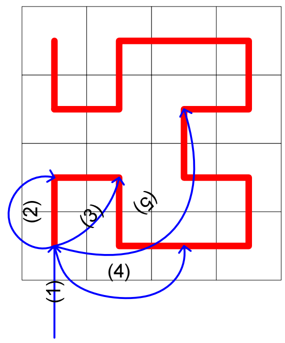

Fig. 4. Expressway construction using landmark clustering. (1) Node N with landmark number 0 gets information about other nodes with the same landmark number and establishes a connection with any one of them; (2) Node N gets information about nodes with landmark number $2^0 = 1$; (3) Node N gets information about nodes with landmark number $2^1 = 2$; (4) Node N gets information about nodes with landmark number $2^2 = 4$; (5) Node N gets information about nodes with landmark number $2^3 = 8$.

bly be introduced due to dimension reduction. (2) Using multiple realities improve the accuracy of landmark clustering at the cost of increased maintenance cost of a separate structure. In the rest of the paper, we assume Hilbert curves are used.

An ordinary node locates the expressway node that it is *closest* to by (i) computing its own landmark number; (ii) using its landmark number as the key to route to the location where information about other nodes that have similar landmark numbers is stored; and (iii) performing a localized flooding within a specified radius starting from the destination location until information for some expressway node is encountered. Localized flooding is possible because information about close-by nodes is stored close to each other on the overlay. In reality, even the localized flooding can be avoided by either replicating the published information to nearby nodes, or build a “condensed map” by storing the information in only a sub region of the default overlay [18].

We are now ready to describe how to build a fully connected expressway using the landmark number stored in the default overlay itself. The basic idea is to have each node connect to a number of other nodes whose landmark numbers are numerically close to its landmark number. This approach is completely decentralized. Figure 4 shows how the nodes with landmark number 0 select their expressway neighbors. The neighbors are selected based on their landmark numbers that are at 2^i distance from the current node's landmark number.[1]

In summary, using landmark numbering ensures that the expressway has good physical proximity. This can reduce the latency in propagating the route advertisements and expressway routing itself. Landmark numbering can be performed repeatedly to reflect changing network conditions.

B. Route Advertisement with Summarization

Each expressway node periodically advertises all the local nodes that are in its physical proximity to its neighboring expressway nodes. Instead of advertising the logical IDs of the nodes directly, we employ a summarization scheme to provide a means to control the amount of routing state maintained at each node (to trade-off between routing performance and routing table size). For CAN, the summarization is based on the notion of virtual grids in the Cartesian space and a numbering scheme explained later. For other overlays such as Pastry or Tapestry, the prefix or suffix of the nodes can be used to summarize the logical space to the same effect.

[1]In our experimentation, we set the distance to 1.1^i to make the expressway better connected.

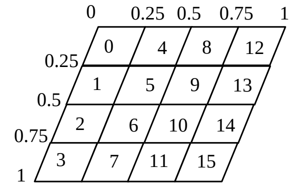

Fig. 5. Summarizing a 2-dimensional Cartesian space using 4^2 grids.

For CAN, the entire d-dimensional Cartesian space is partitioned into m^d virtual grids of equal size, and each virtual grid is assigned a number ranging from 0 to $m^d - 1$, called the virtual grid ID. Each default CAN zone is summarized using the ID of the virtual grid in which the center of the CAN zone falls. Similarly, any point in the Cartesian space is numbered using the same scheme.

Figure 5 illustrates the partitioning of a 2-dimensional Cartesian space using 4^2 grids. A node uses the ID of the grid to which it maps during route advertisement. For example, if a node has a coordinate of $\{0.1, 0.3\}$, then it uses virtual grid ID of 4 during route advertisement.

The algorithm for route advertisement is the same as the standard distance vector algorithm [16]. The differences are that (i) instead of advertising just a node's transport address, we also advertise the virtual grid ID that is used for summarizing nodes; (ii) only expressway nodes participate in route advertisement; (iii) the route advertisement messages are controlled with a time-to-live (TTL) value expressed as a number of expressway-node hops. Having a small number of virtual grids would produce less precise advertised information but the route state that an expressway node needs to maintain becomes smaller. Even when the virtual grid is larger than the zone to be advertised, routing to any zone that belongs to the virtual grid guarantees that the target is inside the virtual grid and can be routed with the default overlay routing in a bounded number of logical hops.

In addition to the default routing table for eCAN, each ordinary node keeps the addresses of the local expressway nodes, whereas expressway nodes maintain route summaries. The number of entries in a route summary is on the order of the number of virtual grids used for summarizing the nodes in the system.

Expressway routing itself can be done in two different ways: using the IP address propagated in route advertisement (direct route), and using the expressway nodes to forward the packets (expressway-node forwarding). The first approach requires slightly more storage space to keep the route summary and relies on IP routing. However, if a node leaves the network, the second approach is less expensive to repair because the grids to summarize the zones does not change even when nodes join and leave the system. In addition, expressway forwarding provides a way to handle multicast efficiently, and possibly deliver performance better than default IP routing. In Section IV, we will compare the two approaches.

C. Routing Algorithm

Routing with an expressway in place is straightforward. When an expressway node receives a packet it performs the following operations:

1) It checks the route summary. If the destination is in the route summary, then it routes the packet to the destination directly.
2) Otherwise, it checks to see if there is any expressway node that is logically closer[2] to the destination than the current node. If it finds such a node, it sets the expressway-used flag in the packet and then forwards it to the expressway node.
3) Otherwise, it sets the expressway-used flag in the packet and uses the default routing (i.e., eCAN).

For non-expressway nodes, the algorithm is the following:

1) If the expressway-used flag is set, use default routing.
2) Otherwise, check to see if the destination is one of its eCAN neighbors. If so, it routes the packet to that neighbor directly.
3) Otherwise, it sends the packet to one of the local expressway nodes.

If the destination is in the route summary, the expressway will route the packet to the destination or to a node that is logically close to the destination. If the expressway routes the packet to a node that is not the destination, from then on only default routing will be used by the algorithm. If it is not in the route summary, the an expressway-used flag will be set on packet, and from there on the algorithm will only use default routing. This guarantees that the packet will reach its destination.

D. Dynamism and Scalability Issues

It is reasonable to expect dynamism in the node membership in a large-scale widely distributed environment. This significantly impacts the routing state maintained by a node. However, we believe that events such as joining and leaving are less frequent in the case of expressway nodes because we choose only relatively stable nodes to act as expressway nodes. For a highly dynamic environment, the benefits of an auxiliary network will be lower than for a static environment. Even caching is less effective in that case and the system will have to rely on the default overlay routing.

As far as route propagation is concerned, the nodes that reside in the virtual grids can change dynamically. How timely this information can be propagated is an open issue. However, we believe it is not critical because it will not affect the correctness of routing. An expressway using direct route will perform at least as well as Brocade even if we decide not to perform route update when an owner of a virtual changes. In this case, the source node sends the packet directly to the expressway node that has information about the target virtual grid. This is in essence, how Brocade works.

In essence, we have used the default overlay as an information exchange repository where the expressway nodes publish information about themselves and all the nodes retrieves that information. This is done in a totally decentralized fashion without any centralized coordination. If a node that hosts this information leaves the system, we leverage the mechanisms provided by the underlying overlay. In addition, we can also refresh this information periodically to keep up with the changing network conditions.

[2]If the expressway is fully connected then we can send the packets to an expressway node that is physically closest to the destination.

IV. EXPERIMENTAL EVALUATION

We evaluate our techniques using simulations with an Internet-like topology and a transit-stub graph produced using GT-ITM [17].

A. Simulation Methodology

We use simulations to compare our expressway approach against several different approaches: the optimal (or shortest-path) routing where a packet is delivered along the shortest path from the source to the destination, eCAN, and a Brocade-like system that constructs a logical auxiliary network using eCAN. From now on, we will refer the Brocade-like system used in our experimentation as *logical auxiliary*. To make the comparison fair, the nodes in eCAN keep roughly the same amount of routing state as our expressway approach, and only expressway nodes participate in high-order routing.

We use eCAN as the basis for the logical auxiliary, because we want to compare against Brocade-like systems in the same setting, i.e., to compare against the idea of using a logical overlay as an auxiliary network for advertising the addresses of the ordinary nodes. In addition, we do not have access to a Brocade simulator. We believe the comparison is fair because both eCAN and Tapestry achieve logarithmic logical routing performance and are topology-aware.

For the logical auxiliary, all nodes except the expressway nodes (supernodes) advertise their IDs and locations in the auxiliary network using summaries of their zones as keys to place the advertisement in the overlay. We evaluated two versions: no caching of advertisements, and caching the information of ordinary nodes along the path of advertisements. To make the comparisons fair, all simulations of the logical auxiliary advertise the IP addresses of the nodes such that once the advertisement is located, direct IP routing will be used. In addition, we also assume nodes know the IP addresses of all other nodes in the same AS, which is the assumption made by Brocade.

The primary performance metric that we use is *stretch* defined as the ratio of accumulated latency in the actual routing path to the shortest-path latency from the source to destination.

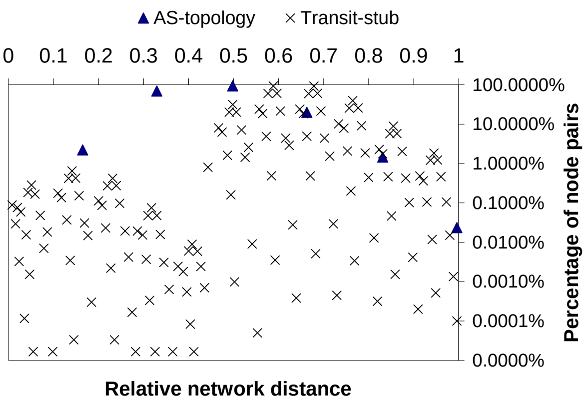

Fig. 6. Latency distribution of the two topologies.

The average routing delay is computed by randomly selecting 10 times as many source destination pairs as there are nodes, and then averaging the routing delay between them.

We evaluate our techniques using two topologies. An Internet-like topology derived from BGP report, and a transit-stub graph produced using GT-ITM [17]. We describe the two topologies below.

For the AS-level topology, we derive the delay of each resolution from the actual number of AS hops taken and the average delay between two AS. We have generated an approximate topology of the Internet by extracting AS connectivity information from BGP reports [10] and computing shortest path distances between all pairs of AS. In our experiments we use 1,000 AS from a total of 13,000 active AS. When nodes in the default overlay are populated, each node is assigned to one of the 1,000 AS. We assume an average inter-AS delay of 100 ms and an average intra-AS delay of 10 ms in the experiments.

To provide a finer grained topology, we also evaluate our techniques using a transit-stub graph produced by GT-ITM. This topology has approximately 10,000 nodes, 228 transit domains, 5 transit nodes per transit domain, 4 stub nodes attached to each transit node, and 2 nodes in each stub domain. The latency is set according to the following rules: 100 ms for cross transit links, 20 ms for links connecting nodes inside a single transit, 5 ms for links connecting a transit node and a stub node, and 2 ms for links connecting nodes inside a single stub. Figure 6 compares the latency distribution of the two topologies. This distribution is computed among all pairs of nodes in the two topologies. From the figure, we can see that the AS-topology is fairly coarse-grained with only small variations in routing delays, where as the transit-stub graph demonstrates some clustering behavior with nodes either far away from each other or relatively close to each other.

In our experiment, we ignore the overhead in processing the packets, as we do not have a real implementation yet. The parameters that are varied in our experiments include the number of nodes, the size of the virtual grid, the TTL, and the percentage of expressway node in the entire node population. Table I summarizes the parameters and their values used in the simulation. We use landmark clustering only as a pre-selection process to identify the nodes that are possibly close to a given node, and use RTT measurements to identify the node that is actually the closest. The row "Extra RTT measurements" gives the additional measurements we perform to identify the actual closest nodes. For example, if the number of nodes we want to identify is 10, then the actual measurements we carry out is $10+20$, and 20 is the number of extra measurements. The row "Fraction of expressway nodes" gives the percentage of expressway node in the entire node population, and 1/10 means that approximately one out of ten nodes will act as an expressway node.

TABLE I

PARAMETERS USED IN THE SIMULATION.

Parameters	Default	Range
# of nodes (n)	-	512 – 8K
TTL	9	1 – 9
Virtual Grids	$(2^{\lceil \log_2 \sqrt{n} \rceil})^2$	$(2^{\lceil \log_2 \sqrt{n} \rceil})^2$ – $(2^{\lceil \log_2 \sqrt{n} \rceil})^2/4$
# of Landmarks	15	-
Extra RTT measurements	20	-
Fraction of expressway nodes	1/10	1/1 – 1/64
Routing	Direct	direct, forwarding

For the AS-topology, we control the ratio of expressway nodes by changing the number of AS that participate in the overlay. If the fraction is 1/8 for 512 nodes, the number of AS will be 64, and there will be roughly 8 nodes in the same AS. For the transit-stub graph, we randomly select nodes to act as expressway node according until we get the desired fraction of expressway nodes.

B. Experimental Results

Figure 7 shows the effect of varying the number of nodes in the system from 512 to 8K. We fix the TTL, the number of virtual grids, and the number of landmarks using the default values shown in Table I.

The curve labeled Exp (AS) constructs the expressway using AS-level topology information and employ direct route. The curve labeled Exp (landmark) employs the landmark-clustering scheme to construct a fully connected expressway, whereas the curve labeled Exp(AS+landmark) constructs an expressway using landmark clustering in addition to applying AS neighboring information. The curves labeled "logical auxiliary" show the performance of a systems that use a secondary overlay network. The "logical auxiliary (advertising)" curve also caches the IP addresses of ordinary nodes along the advertising path. As we do not have any AS information for the transit-stub topology, we cannot show any results for Exp (AS) and Exp(AS+landmark) using that topology.

From Figure 7 we observe the following. First, constructing an expressway by using AS neighboring information can produce good routing performance. On average the routing performance is about 1.09 times shortest-path routing with individual measurements ranging from 1.04 to 1.16 times shortest-path routing.

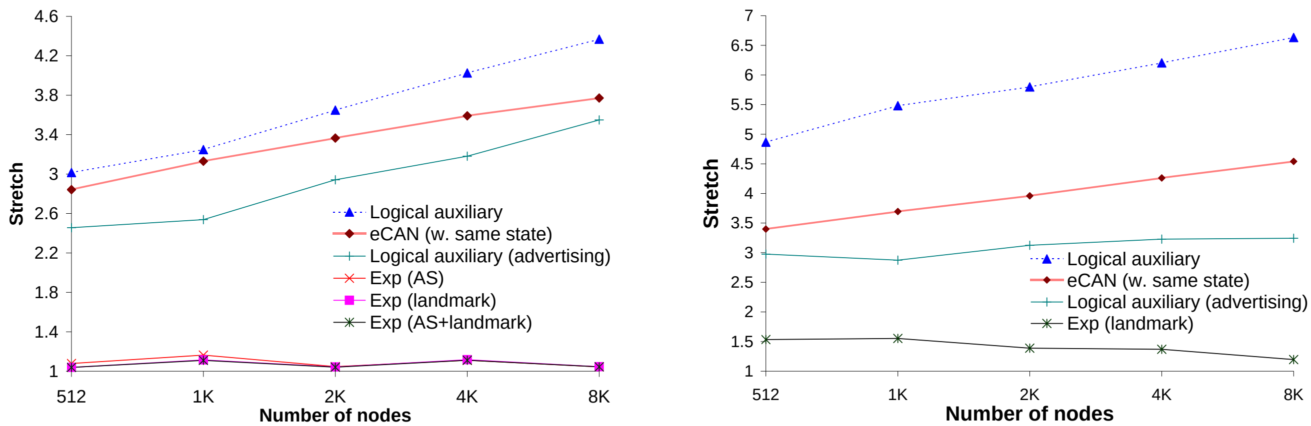

Fig. 7. Comparison of various approaches using the AS topology to the left and the transit-stub topology to the right.

Second, expressway routing using landmark clustering produces close to optimal routing performance for the AS topology. On average the routing performance is about 1.07 times shortest-path routing with individual measurements ranging from 1.04 to 1.12. The routing performance for the transit-stub graph is slightly worse than that for the AS topology with individual measurements ranging from 1.20 to 1.55 times shortest-path routing. The performance differences between the two different topologies are due to the fact that for the transit-stub graph, ordinary nodes that are close to the same expressway node do not establish direct connection with each other. In addition, for the AS topology, we assume that there is always one expressway node in an AS, and an ordinary node can reach an expressway node within 10 ms. As we will see later, increase the percentage of expressway node in the entire node population can improve expressway routing performance.

Third, the performance of eCAN with similar total routing state is comparable to that of the logical auxiliary. In fact, eCAN performs consistently better than the logical auxiliary without advertisement. This is because the eCAN we use in our experiments keeps slightly more routing state than that of the logical auxiliary and we have employed a simple heuristic to balance physical and logical routing delay when selecting next hops for a particular routing destination [18]. The performance differences are more prominent for the large transit-stub graph. We hypothesize that the additional routing state kept by eCAN and the heuristic work better the larger transit-stub topology.

It should be noted that if we cache the route advertisements everywhere on the logical auxiliary network, it can obtain similar performance to shortest-path routing and expressway routing using landmark clustering. The key difference comes from the fact that the expressway closely approximates the physical network, whereas the logical auxiliary will always be a logical overlay. As a result, using expressway for multicast would be more efficient than using a logical auxiliary. In addition, when the auxiliary network closely approximates the underlying physical network, it would be more efficient to propagate the routing state along the physical overlay than along a logical overlay.

TABLE II

THE AMOUNT OF ROUTING STATE.

# of Nodes	# of Grids	Routing table (# entries)		
		Expressway (Avg)	eCAN (Avg)	
			AS topology	Transit-stub
512	32x32	1K	36	29
1024	32x32	1K	39	31
2048	64x64	4K	41	33
4096	64x64	4K	43	35
8192	128x128	16K	45	38

Figure 8 compares direct routing with expressway-node forwarding for the two different topologies. From the figures, we can see expressway-node forwarding does not degrade performance by much and it provides the advantages mentioned in Section III-B. The performance improves with more nodes in the system. When there are more expressway nodes, the expressway has a better chance to approximate the underlying physical network. There is a larger gap between the performance of direct routing and expressway-node forwarding for the transit-stub graph because that the transit-stub graph is a larger topology than the AS topology we used. When there are more nodes in the system, the number of backbone nodes accounts for only a small percentage of all nodes in the system. This makes it hard for the expressway to approximate the underlying physical topology. There is a potential for expressway-node forwarding to perform better than default IP routing, as shown by RON [19].

To give a feel for the cost paid to achieve near optimal routing performance, Table II shows the routing state that each node needs to keep for expressway routing using landmark clustering for the two topologies. Note that only expressway nodes need to maintain route summaries. With modern computer systems, we believe keeping this amount of state should not be an issue. In our experiments, we assume that each AS has at least one expressway node. This can vary in an actual implementation. If there is no expressway node in an AS, we can always rely on landmark numbering to locate an expressway node in an AS that is close-by.

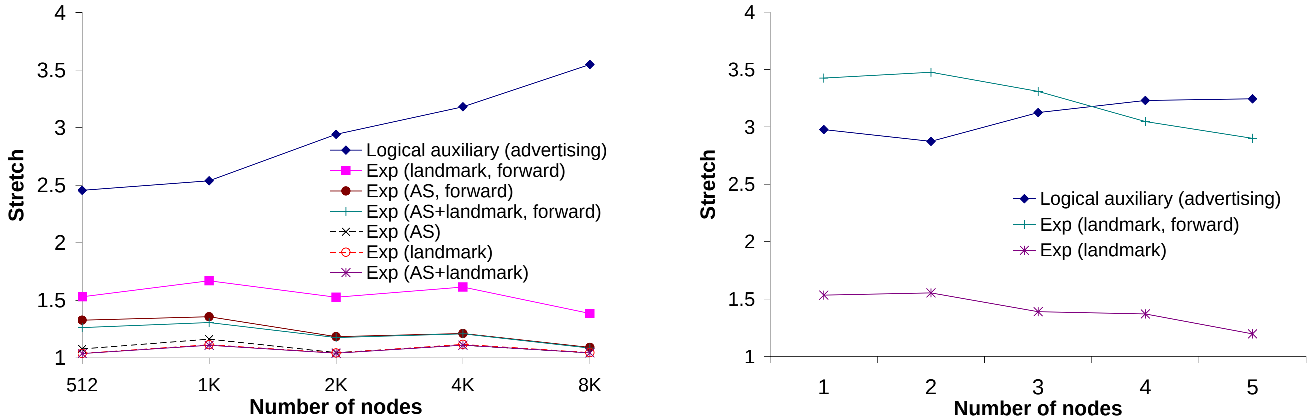

Fig. 8. Comparisons of direct route and expressway-node forwarding using the AS topology on the left and the transit-stub topology on the right.

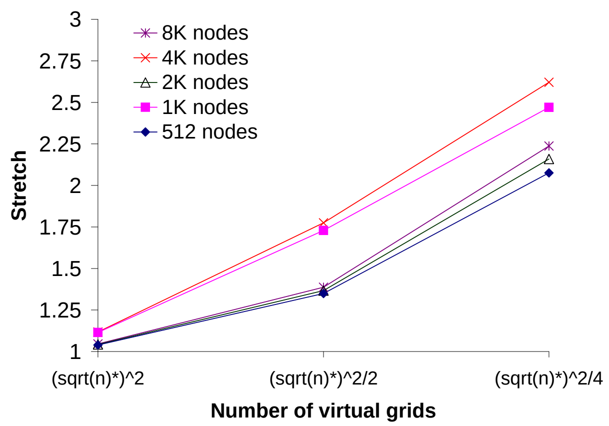

Fig. 9. Varying the number of virtual grids. sqrt(n)* represents the smallest power of 2 number that is greater than or equal to $\sqrt{n}$, where n is the number of nodes, i.e. sqrt(n)* $= 2^{\lceil \log_2 \sqrt{n} \rceil}$.

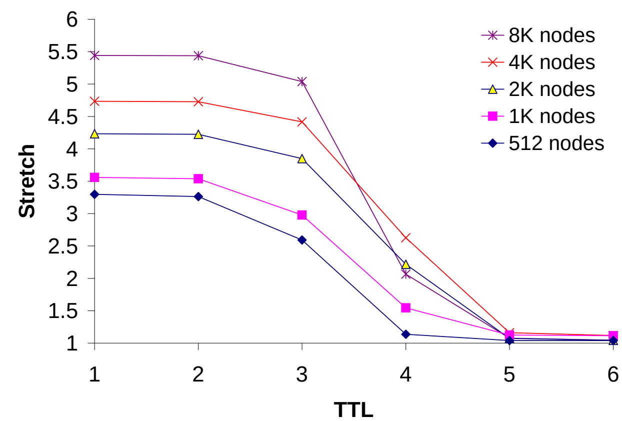

Fig. 10. Effect of varying TTL for route advertisement.

The trade-off is that when there are more expressway nodes the non-expressway nodes need to keep less routing state. The larger the number of expressway node, the larger the likelihood that an ordinary node can find an expressway node that is nearby, and hence the better the routing performance. The state each expressway node needs to keep depends on the number of nodes in the system and the number of virtual grids used. However, the number of packets each expressway node needs to handle depends only on the number of expressway nodes in the system. As the number of expressway nodes decreases the load on them increases.

To understand the effect of varying different parameters that affect routing performance, we first show the results for varying the number of virtual grids used in a route summary. We do this only for the AS topology and would expect the transit-stub topology to exhibit similar behavior. The results in Figure 9, show that reducing the number of grids can significantly affect routing performance. However, the effect is only proportional to the square root of the number of grids. This indicates that we can obtain a relatively bigger saving in routing state with a small sacrifice in performance. In our experiments, when the virtual grid is occupied by multiple nodes, we assume that default overlay routing will be used once the packet reaches the node that represents the destination virtual grid. To address this problem a direct route can be used by, for example, employing some kind of local gossiping protocol for the nodes in the same virtual grid to exchange their IP addresses. Consequently, the performances will be at most twice as much as shortest-path routing.

Figure 10 shows the results for varying the TTL parameter again for the AS topology. As can be seen, the TTL value can significantly affect routing performance if not chosen correctly. In certain cases, increasing the TTL by one can almost cut routing latency by half. In our experiment, a TTL of 5 is enough to bring the routing performance to close to optimal.

As we mentioned earlier, varying the fraction of expressway nodes can also affect routing performance. When there are more expressway nodes in the system, there is a larger likelihood that an ordinary node can find an expressway node that is nearby. To demonstrate this effect, we vary the fraction of expressway node from 1/1 to 1/64, and 1/64 means that approximately every 64 nodes there will be an expressway node. Figure 11 shows the results using the transit-stub topology and validates our hypothesis. An interesting thing to note is that the performance of the logical auxiliary degrades as the number of supernodes increases. This is because that with a larger secondary overlay, it will take longer time to retrieve

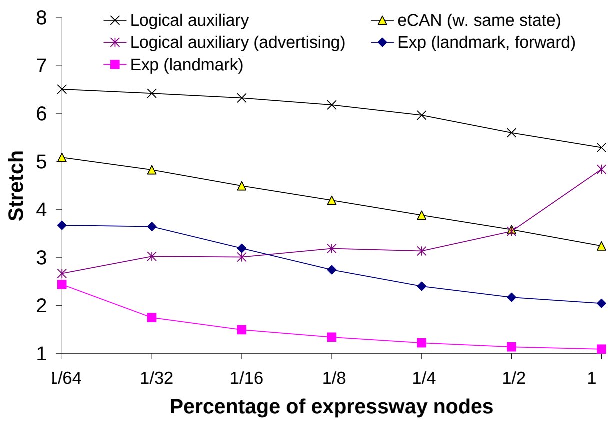

Fig. 11. Effect of varying the ratio of expressway nodes in the system.

the published information about the ordinary nodes. However, when the number of nodes in the auxiliary is small, the load on each of those nodes will be higher and it will take longer time for an ordinary node to reach the closest supernode. In the case of the expressway, on the other hand, the more expressway nodes, the better it can approximate the underlying physical network.

Another parameter that affects routing performance is the number of landmarks used for generating landmark numbering. The more landmarks, the less amount of false clustering and the less RTT measurements are needed. In our experiments, we employ a technique that stores the published information only in a sub region of the default overlay to increase the chance that a node can always find information of expressway nodes that are nearby [18]. We did not see a significant difference in the routing performance when the number of landmarks is greater than 5 and is varied.

V. Related Work

Techniques to exploit topology information in overlay routing can be subdivided into three categories [14]: *geographic layout*, *proximity routing*, and *proximity neighbor selection* techniques.

With geographic layout, such as Topologically-Aware CAN [13], when a new node joins the overlay, it joins a node that is close to it in IP distance. This can result in uneven node distribution and holes in the Cartesian space, which can create overhead in data placement.

Proximity routing is employed in Chord [8]. With proximity routing, physical topology is not taken into consideration when constructing the overlay. Instead, it employs heuristics that use many logical hops with small latency instead of fewer logical hops with large latency. However, the choices are limited to entries present in the routing tables.

Proximity neighbor selection is employed in Pastry [9] and Tapestry [11]. Routing table entries are selected according to the proximity metric among all nodes that satisfies the constraints of the logical overlay. For instance, in Pastry, the constraint is the node ID prefix. In these systems, however, nodes cannot independently discover their neighbors in a decentralized fashion [7]. For instance, in Pastry, a new node X first has to contact a node A that is physically close to it. It asks node A to route to node X's ID. Along the way, the routing table for X is constructed by picking up the level i routing table from the i^{th} node encountered. This approach assumes the topology satisfies triangle inequality which may not hold for Internet. In fact, a study by Castro *et al.*[14] has shown that the proximity approximation is much worse when using the Mercator topology that is based on the real measurements of the Internet. Further, they rely on expanding-ring search for locating the physically closest node at node join, which according to Xu *et al.*[22] requires a considerable amount of message (hundreds to thousands) exchanges to obtain reasonable results.

In Brocade [15], a secondary overlay network of supernodes is used to improve the routing distance. These supernodes are the nodes that are situated near the network access points such as routers and gateways. Nodes in the default network establish direct connections with a supernode that is nearby. Supernodes advertise the nodes that are connected to them as objects served by the supernode in the secondary overlay. Routing from node A to D in the default network involves three steps: locating a supernode B locally, routing to the supernode C that stores the object D in the secondary overlay, and hopping from C to D.

Though Brocade produces some reasonable improvements from the default case, it pushes the problem to an auxiliary network of a smaller size. Brocade faces a dilemma in choosing the appropriate number of supernodes: if the number of supernodes is large, the logical overlay routing cost gets higher; if the number of nodes in the secondary network is small, an ordinary node needs to keep more state about the addresses of other nodes that are linked to the same supernode. This can be an issue in a dynamic environment. An expressway closely approximates the physical network, making performance independent of the size of the network.

In our simulation study, Brocade-like systems can approach optimal routing performance only when aggressive caching is used. Even then, we expect expressway routing to outperform Brocade with respect to multi-cast performance. In addition, our expressway approach opens up the possibility of delivering routing performance better than default-IP routing.

VI. Conclusions

In this paper, we described generic techniques to construct an auxiliary network for any distributed hash-table based overlay to take advantage of the inherent node heterogeneity that exists in the physical network.

Our approaches use proximity information to construct an auxiliary network called an expressway. We use two different techniques to derive proximity information. The first approach uses an AS-level topology that is derived by processing BGP reports. The second approach uses a novel landmark clustering technique that clusters nodes that have similar latencies to well-known landmarks. As a result, we are able to reconcile the conflict of presenting the applications with a homogeneous structured overlay to simplify management, while at the same time exploiting the inherent heterogeneity of the underlying physical network.

We conducted a detailed simulation study using two topologies. The simulation results show that our approaches can achieve close to optimal routing performance. For the AS-topology, our expressway based on landmark-clustering approach yields 1.07 times optimal routing performance on average, while the approach that uses AS-neighbor information yields about 1.09 times optimal routing performance. For the transit-stub topology, our landmark-clustering technique yields 1.4 times optimal routing performance on average.

Although, we have evaluated expressway using eCAN as the default overlay, the techniques proposed in this paper are generic and directly applicable to other DHT-based systems.

ACKNOWLEDGMENT

We thank Christos Karamanolis, Ira Greenberg and Chunqiang Tang for their valuable feedback on this paper.

REFERENCES

[1] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "Oceanstore: An architecture for global-scale persistent storage," in *17th ACM Symposium on Operating Systems Principles (SOSP '00)*, November 2000, pp. 190–201.

[2] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," in *18th ACM Symposium on Operating Systems Principles (SOSP '01)*, October 2001, pp. 202–215.

[3] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *12th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, May 2002.

[4] V. N. Padmanabhan and K. Sripanidkulchai, "The case for cooperative networking," in *1st International Workshop on Peer-to-Peer Systems (IPTPS)*, March 2002.

[5] C. Tang, Z. Xu, and M. Mahalingam, "pSearch: Information retrieval in structured overlays," in *1st Workshop on Hot Topics on Networks (HotNets-I)*, October 2002.

[6] A. Keromytis, V. Misra, and D. Rubenstein, "Sos: Secure overlay services," in *2002 ACM SIGCOMM Conference*, August 2002, pp. 61–72.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," in *2001 ACM SIGCOMM Conference*, 2001, pp. 161–172.

[8] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable Peer-To-Peer lookup service for internet applications," in *2001 ACM SIGCOMM Conference*, 2001, pp. 149–160.

[9] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, November 2001, pp. 329–350.

[10] *Telestra.net*, http://www.telestra.net.

[11] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," UC Berkeley, Tech. Rep. UCB/CSD-01-1141, April 2001.

[12] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson, "The end-to-end effects of internet path selection," in *1999 ACM SIGCOMM Conference*, 1999, pp. 289–299.

[13] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *IEEE INFOCOM 2002*, June 2002.

[14] M. Castro, P. Druschel, Y. Hu, and A. Rowstron, "Exploiting network proximity in distributed hash tables," in *International Workshop on Future Directions in Distributed Computing (FuDiCo 2002)*, 2002.

[15] B. Y. Zhao, Y. Duan, L. Huang, A. Joseph, and J. Kubiatowicz, "Brocade: Landmark Routing on Overlay Networks," in *1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, 2002.

[16] M. E. Steenstrup, Ed., *Routing in Communications Networks*. Englewood Cliffs, NJ: Prentice Hall, 1995, ch. Distance Vector Routing, ISBN: 0130107522.

[17] *GT-ITM*, http://www.cc.gatech.edu/projects/gtitm/.

[18] Z. Xu and Z. Zhang, "Building Low-maintenance Expressways for P2P Systems," Hewlett-Packard Labs, Palo Alto, CA, Tech. Rep. HPL-2002-41, 2002.

[19] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *18th ACM Symposium on Operating Systems Principles (SOSP)*, October 2001, pp. 131–145.

[20] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan, "Detour: a Case for Informed Internet Routing and Transport," *IEEE Micro*, vol. 19, no. 1, pp. 50–59, January 1999.

[21] T. Asano, D. Ranjan, T. Roos, E. Welzl, and P. Widmaier, "Space Filling Curves and Their Use in Geometric Data Structures," *Theoretical Computer Science*, vol. 181, no. 1, pp. 3–15, July 1997.

[22] Z. Xu, C. Tang, and Z. Zhang, "Building topology-aware overlays using global soft-state," HP Labs, Tech. Rep. HPL-2002-281, September 2002.